Quantitative Analysis of Subcellular Biomechanics and Mechanotransduction

by

Jan Lammerding

B.E., Thayer School of Engineering, Dartmouth College, 1997

Dipl. Ing., Mechanical Engineering, Rheinisch Westfälische Technische Hochschule Aachen, 1999

Submitted to the Biological Engineering Division
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy in Bioengineering
at the
Massachusetts Institute of Technology

June 2004

Signature of Author: .....
Jan Lammerding
Biological Engineering Division

Certified by: .....
Roger Kamm
Professor of Biological Engineering
and Mechanical Engineering
Thesis Supervisor

Certified by: .....
Richard Lee
Associate Professor Harvard Medical School
Thesis Supervisor

Accepted by: .....
Alan J. Grodzinsky
Professor of Biological Engineering
Chairman, Committee for Graduate Students

This doctoral thesis has been examined by a committee of the Biological Engineering Division as follows:

Thesis Supervisor, Chairperson, Graduate Thesis Committee: .....

Richard T. Lee
Professor Harvard Medical School

Thesis Supervisor, Committee Member: .....

Roger D. Kamm
Professor of Biological Engineering

Thesis Committee Member: .....

Peter T. So
Professor of Biological Engineering

Quantitative Analysis of Subcellular Biomechanics and Mechanotransduction

by

Jan Lammerding

Submitted to the Biological Engineering Division
on May 21, 2004 in Partial Fulfillment of the Requirements for the Degree of
Doctor of Philosophy in Bioengineering

Abstract

Biological cells such as endothelial or muscle cells respond to mechanical stimulation with activation of specific intracellular- and extracellular signaling pathways and cytoskeletal remodeling, a process termed mechanotransduction. Intracellular mechanosensors are thought to be activated by conformational changes induced by local cellular deformations. Since these mechanosensors have been speculated to be located in several cellular domains including the cell membrane, the cytoskeleton, and the nucleus, it is necessary to achieve a detailed understanding of subcellular mechanics. In this work, we present novel methods to independently quantify cytoskeletal displacements, mechanical coupling between the cytoskeleton and the extracellular matrix, and nuclear mechanics based on high resolution tracking of cellular structures and receptor bound magnetic beads in response to applied strain or microscopic forces. These methods were applied to study the effects of several human disease associated mutations on subcellular mechanics and to examine the interaction between known protein function and specific changes in cellular mechanical properties and mechanotransduction pathways.

Initial experiments were targeted to the role of membrane adhesion receptors. Experiments with cells expressing a mutant form of the integrin-associated molecule tetraspanin CD151 revealed that CD151 plays a key role in selectively strengthening $\alpha 6\beta 1$ integrin-mediated adhesion to laminin-1. We then studied cytoplasmic behavior using cells from mice with an αB-Crystallin mutation (R120G) that causes desmin-related myopathy. These studies showed impaired passive cytoskeletal mechanics in adult mouse cardiac myocytes. Finally, we studied cells deficient in the nuclear envelope protein lamin A/C and showed that lamin A/C deficient cells have increased nuclear deformation, defective mechanotransduction, and impaired viability under mechanical strain, suggesting that the tissue specific effects observed in laminopathies such as Emery-Dreifuss muscular dystrophy or Hutchinson-Gilford progeria may arise from varying degrees of impaired nuclear mechanics and transcriptional regulation.

In conclusion, our methods provide new and valuable tools to examine the role of subcellular biomechanics on mechanotransduction in normal and mutant cells, leading to improved understanding of disease mechanisms associated with altered cell mechanics.

Thesis Supervisors: Roger D. Kamm, Richard T. Lee

Acknowledgements

First and foremost I would like to thank my advisors and my thesis committee for their outstanding guidance, support and inspiration. I'd like to thank Rich for giving me the opportunity to work in his lab and for teaching me so many aspects of what it takes to succeed in science and academia, from adding extra controls to experiments to how to write scientific papers and respond to reviewers comments. These lessons will guide me through the rest of my career. I'd like to thank Roger for making all this possible in the first place, from introducing my to the Biological Engineering Program when I first applied to MIT five years ago and for subsequently taking me on as a student and attracting me to the field of biomechanics and mechanotransduction. I'd like to thank Peter, for reminding me and all of us how much fun and excitement it can be to look at single cells through a microscope and for always coming up with new imaging ideas and techniques for our research projects.

I would also like to thank all members of the Lee and Kamm laboratories, especially Hayden for teaching me the fine art of cell culture, Christian, Tomo and Jun for introducing me to molecular biology, Mihaela for all her great work and help with the mice, and Janet for all her help and efforts in the Lamin/Emerin project. None of this work would have been possible without you. This is also true for our outstanding collaborators who provided cells, mice, and priceless expertise to this work. For this I express my deepest gratitude to Martin Hemler, Colin Stewart, Howard Worman, Jeffrey Robbins and Jon Seidman.

Last but not least, I would like to thank my family, Christel, Jo, Tim, and Yu, for their endless support, love, care, and encouragement over all these years, and my friends for all their company, help or just welcome distraction. You all contributed to make this an enjoyable and fun journey.

This thesis is dedicated to the loving memory of my mother, Marianne Lammerding.

Table of Contents

1 Introduction 9
1.1 Mechanotransduction 9
1.1.1 Mechanotransduction and hypertrophy in cardiac myocytes 9
1.1.2 Cellular signaling in response to mechanical stimulation 11
1.1.3 Mechanosensors in cardiac myocytes 17
1.1.4 Mechanotransduction in other cells types 22
1.2 Cellular mechanics 24
1.3 Thesis objective 29
1.4 References 34
2 Experimental and analytical methods 40
2.1 Magnetic trap design and calibration 40
2.2 Strain device for microscopic observations 43
2.3 Particle tracking algorithm 45
2.4 Strain analysis program 47
2.5 Cell type specific analysis 50
2.6 References 61
3 Tetraspanin CD151 regulates α6β1 integrin-dependent mechanical force transduction 62
3.1 Introduction 62
3.2 Materials and methods 63
3.3 Results 66
3.4 Discussion 70
3.5 References 82
4 Experimental verification of a three-dimensional viscoelastic finite element model for cell deformation 84
4.1 Introduction 84
4.2 Materials and methods 85
4.3 Results 88
4.3.1 Numerical model 88
4.3.2 Cell experiments 91
4.4 Discussion 93
4.5 References 104
5 Quantitative measurements of passive mechanical properties in normal and hypertrophic adult mouse cardiac myocytes 105
5.1 Introduction 105
5.2 Materials and methods 108
5.3 Results 111
5.4 Discussion 114
5.5 References 127
6 Lamin A/C deficiency causes defective nuclear mechanics and mechano-transduction 128
6.1 Introduction 128
6.2 Materials and methods 129

6.3 Results 133
6.4 Discussion 137
6.5 References 150
7 Conclusions and outlook 153
7.1 References 158

Appendix A Magnetic trap design draft 159
Appendix B Nuclear strain as an indicator of nuclear mechanical properties 160
Appendix C Monocular 3-D magnetic bead microrheometry 162
Appendix D Isolation of adult mouse cardiac myocytes 167
Appendix E Mouse colony maintenance 181
Appendix F Matlab programs 185
Appendix G LabView programs 282

List of Tables and Figures

Table 2.1 Strain device calibration 52
Table 2.2 Spatial calibration 53
Table 2.3 Particle tracking resolution for synthetic images 54
Table 2.4 Particle tracking resolution under experimental conditions 55

Figure 1.1 Role of mechanotransduction in cardiac hypertrophy 32
Figure 1.2 Signal transduction pathways involved in the cellular response to mechanical stimulation 33
Figure 2.1 Magnetic trap set-up 56
Figure 2.2 Magnetic trap operating temperature curve 57
Figure 2.3 Strain device design 58
Figure 2.4 Screenshot of the particle tracking program 59
Figure 2.5 Screenshot of the strain analysis program 60
Figure 3.1 CD151-α6β1 integrin complex contributes to NIH3T3 cell assembly into cables 74
Figure 3.2 Mutant CD151 and CD151-WT effects on static cell adhesion and cell surface expression 75
Figure 3.3 Magnetic trap characterization 76
Figure 3.4 Detachment of laminin and fibronectin-coated beads from NIH 3T3 transfectants 77
Figure 3.5 Bead detachment effects of anti-α6 antibody and bead attachment by laminin and fibronectin beads 78
Figure 3.6 CD151 mutant effects are time-dependent and ligand dependent 79
Figure 3.7 Magnetic trap calibration 80
Figure 3.8 Bead displacement for laminin-coated and fibronectin-coated beads 81
Figure 4.1 General model geometry 98
Figure 4.2 Representative magnetic bead displacement for step force application 99
Figure 4.3 Cross-sectional view of monolayer 100
Figure 4.4 Membrane displacement in the forcing direction 101
Figure 4.5 Bead center displacement versus time 102
Figure 4.6 Bead center displacement as a result of force applied to the bead versus time 103
Figure 5.1 Anisotropic material behavior in adult mouse cardiac myocytes 119
Figure 5.2 Displacement map at onset of sinusoidal force application 120
Figure 5.3 Single sarcomere displacement analysis 120
Figure 5.4 Cell width measurements for mutant and wild-type cardiac myocytes 121
Figure 5.5 Magnetic bead displacement amplitude and phase lag are impaired in R120G-CryAB but not in D7-des mutants 122
Figure 5.6 Cytoskeletal displacement measurements 124

Figure 5.7 **R120G-CryAB mutants have altered cytoskeletal mechanics** 125
Figure 5.8 **Adult mouse cardiac myocyte imaged using bright field and fluorescence microscopy** 126
Figure 6.1 **Nuclear mechanics is impaired in lamin A/C-deficient cells**.....142
Figure 6.2 **Cytoskeletal stiffness is reduced in lamin A/C deficient cells** 143
Figure 6.3 **Nuclear fragility is increased in lamin A/C deficient cells** 145
Figure 6.4 **Impaired mechanotransduction in lamin A/C deficient cells** 147
Figure 6.5 **Defective NF-κB signaling in lamin A/C deficient cells** 149
Figure 7.1 **Nuclear mechanics in wild-type, emerin deficient and lamin A/C deficient primary mouse embryo fibroblasts** 157

1 Introduction*

1.1 Mechanotransduction

Mechanotransduction can be defined as the cellular (signaling) response to mechanical stimulation. Mechanotransduction is not limited to cells specialized for mechanosensing, such as hair cells in the inner ear, but is found in almost all cell and tissue types. Although the molecular mechanisms of mechanotransduction remain incompletely defined, it is clear that mechanotransduction plays a critical role in maintaining physiological cell function, particularly in cells continuously subjected to mechanical stimulation, such as endothelial cells exposed to fluid shear stress or cardiac and skeletal muscle cells in mechanically active contractile tissues. Interestingly, many different cell types respond in similar fashion to mechanical stimulation, whether it is exposure to shear stress or applied stress or strain, indicating that certain common mechanotransduction pathways are likely shared between various cell types.

Experiments performed in this work focus on mechanotransduction signaling in response to stress and strain and on diseases that arise from mutations in structural proteins that predominantly affect skeletal and cardiac muscle. Therefore, the following review will focus on mechanotransduction in cardiac myocytes, but many similar pathways and mechanisms can be found in other cell types, e.g. endothelial cells exposed to fluid shear stress or chondrocytes in compressed cartilage. Some of the specific responses of those cells will be discussed at the end of this section. Excellent reviews for the molecular basis of mechanotransduction can be found in Hamill & Martinac (1), Gillespie & Walker (2) or Davies (3).

1.1.1 Mechanotransduction and hypertrophy in cardiac myocytes

Cardiac myocytes react to diverse mechanical demands with a multitude of transient and long term responses to normalize the cellular mechanical environment. Mechanical stress in the myocardium can be elevated by tissue damage following myocardial infarction,

* Sections of this chapter have been published in the *Annals of the New York Academy of Science*, Vol. 1015, May 2004 (in press).

excessive load (such as hypertension), or intrinsic defects such as mutations in genes encoding sarcomere proteins. Through several pathways that mediate mechanotransduction, cells sense increased strain levels and respond with a broad variety of changes including cellular hypertrophy, i.e. cellular growth through addition of new sarcomeres without proliferation. Depending on the mechanical loading conditions, sarcomere deposition can occur in parallel, which may lead to concentric hypertrophy, or in series, which may lead to dilation of the ventricle (Fig. 1.1). In the former case, generally associated with pressure overload (e.g. due to hypertension or aortic stenosis) parallel sarcomere deposition leads to increased cross-sectional myocyte area and increased ventricular wall thickness. In the latter case, often associated with volume overload (e.g. mitral regurgitation), cells elongate and the ventricle dilates with only a small increase in wall thickness but an increase in overall mass.

Traditionally, cardiac hypertrophy has been interpreted as the long term response to increased loading conditions in an attempt to restore wall stress to normal levels (4, 5). The hypertrophic response is initially beneficial as it partially compensates for the decreased force generation and normalized ventricular wall stress. However, in the long term, the hypertrophic response leads to cardiac failure. One reason is that hypertrophy itself can further destabilize local mechanics. Several reports indicate that hypertrophic tissue has altered contractile (active) and relaxation (passive) mechanics, further destabilizing the mechanical imbalance between contractile forces in the cardiac tissue and the applied hemodynamic load. Additionally, it appears that the same molecular response that initiates the hypertrophy, i.e. trophic and mitotic growth factors, leads to the activation of genes that are more commonly expressed in embryonic myocardium, altering the cellular phenotype and resulting in myocyte structural disarray, altered calcium dynamics and increased interstitial collagen synthesis (6). Further highlighting the importance of mechanotransduction in cardiac hypertrophy, mechanical unloading of the heart through left ventricular assist devices (LVAD) leads to a reduction in myocyte volume accompanied by improved cardiac and myocyte contractility (7). Experiments with heterotopic abdominal heart transplants in mice confirm a decrease in cell volume and increased fractional shortening and calcium transients in myocytes from unloaded

hearts(8), even though histological features do not completely normalize (9). The molecular mechanisms that control hypertrophy are now slowly emerging, and understanding these mechanisms and how they relate to cardiac compensation and failure are critical to developing new treatments for heart failure (5, 10, 11).

1.1.2 Cellular signaling in response to mechanical stimulation

Cardiac myocytes respond to mechanical stimulation with rapid induction of immediate-early genes such as *c-fos*, *c-jun*, *egr-1* and *c-myc* that are transcription factors. Subsequently, additional changes in gene expression occur, eventually leading to an increased rate of protein synthesis. The long term changes in gene expression include the transition of some sarcomeric proteins to their fetal forms, such as cardiac α-actinin to skeletal α-actinin, α-myosin heavy chain (MHC) to β-MHC in rodents (12), and the re-expression of atrial natriuretic peptide (ANP) in the ventricles. However, it is important to point out that the notion of a single genetic program might be oversimplified, as hypertrophy can occur even without increased levels of individual fetal genes, and the hypertrophic response consists of an orchestrated response between independent and cross-talking signaling pathways (Fig. 1.2). The sections below give a brief overview of signaling pathways involved in stress or strain-induced signaling in cardiac myocytes. Please refer to Molkentin & Dorn (13), Ruwhof & van Laarse (12), and Sadoshima & Izumo (5), for excellent reviews of signaling pathways that regulate cardiac hypertrophy.

Guanine nucleotide-binding proteins. Heterotrimeric guanine nucleotide binding proteins (G-proteins) are associated with cell surface receptors, and several functional classes of G-protein coupled receptors play an important role in the heart. The heterotrimeric G-proteins consist of separate G_α and $G_{\beta\gamma}$ subunits that are dissociated upon receptor mediated exchange of GDP to GTP on the G_α subunit, allowing the subunits to interact with a variety of downstream molecules. G_α subunits $G_{\alpha s}$ and $G_{\alpha i}$ interact predominantly with adenyl cyclase, which catalyzes the synthesis of the second messenger cAMP. $G_{\alpha q}$ activates phospholipase C (PLC). Activated PLC can hydrolyze phosphatidylinositol-bis-phosphate (PIP_2) into inositol-triphosphate (IP_3) and diacylglycerol (DAG), which act as second messengers and cause calcium release from

intracellular storage sites and protein kinase C (PKC) activation, respectively (12, 14). In addition to the action of the G_{α} subunit, free $G_{\beta\gamma}$ subunits can directly activate MAPK, PI3K and Ras signaling in the heart (15-17). Several studies in cultured cardiac myocytes as well as in transgenic animals suggest an important role of $G_{\alpha q}$ as a transducer of cardiac hypertrophy (13). Stretch-induced G-protein activation occurs in cultured neonatal rat fibroblasts (18). The activation of $G_{\alpha q}$ and $G_{\alpha i1}$ subunits occurs within 1 min of strain application and is modulated by strain rate and magnitude. A recent study used adenovirus infected cultured neonatal cardiac myocytes and transgenic mice with enhanced $G_{\alpha q}$ signaling to evaluate the role of G_q signaling in cardiac hypertrophy and apoptosis (19). They found that moderate levels of $G_{\alpha q}$ overexpression resulted in hypertrophic growth, while sustained high level activation of $G_{\alpha q}$ led to apoptotic cardiomyocyte death and heart failure. Experiments comparing hypertrophy and cardiac function between wild type and genetically engineered mice that either specifically inhibit G_q-mediated signaling or that lack endogenous norepinephrine or epinephrine suggest that the development of cardiac hypertrophy and normalization of ventricular wall stress are not necessary to preserve cardiac function under pressure overload. In contrast, the genetically modified mice had less hypertrophy and prolonged survival. MAPK activation was significantly increased in wild type cells, but not in the genetically modified animals (20, 21). These results point out an important role in PI3 and MAPK signaling for hypertrophy. Furthermore, these findings indicate that hypertrophic response is neither beneficial nor necessary to maintain cardiac function.

In addition to the heterotrimeric G-proteins, the low-molecular weight GTPases Ras, RhoA, and rac-1 have been implicated in the regulation of cardiac hypertrophy (13). Ras activation can activate all three MAPK branches (ERK, JNK, p38) as well as Raf-1, PI3K and indirectly Rho and thus connects the small GTPases and heterotrimeric G-protein pathways.

Mitogen activated protein kinase. Mitogen activated protein kinases (MAPKs) are serine/threonine kinases that phosphorylate and activate nuclear substrates as well as other kinases (12). MAPKs are the final component of the MAPK cascade that consists of

the MAP kinases, the MAPK/ERK kinases (MEK) and the MEK kinases (MEKK). The MAPKs can be divided into three subfamilies, (*i*) the extracellular-regulated kinase (ERK), (*ii*) the c-Jun N-terminal kinase (JNK), and (*iii*) the p38 MAP kinase. In the heart, ERK1 is the most abundant form of ERK. The ERK pathway can be activated by ligand binding to G-protein coupled receptors and also by mechanical stimulation. ERK1/2 activation increases expression of c-fos and skeletal α-actin (14, 22), two proteins commonly expressed in cardiac hypertrophy. JNK activation occurs in cultured cardiac myocytes in response to mechanical stretch as well as in load-induced cardiac hypertrophy, myocardial infarction and heart failure (13, 23). Recent evidence based on experiments with dominant negative MEKK4, an upstream kinase necessary for JNK activation, strongly suggests JNK as an essential regulator of cardiac hypertrophy in cultured cells and in the adult heart (13, 24). In stretched cardiac myocytes, activation of JNK is relatively slow (15 min) compared to activation of ERK (1-5 min)(5). In cardiac myocytes, activation of p38 can be induced by mechanical stretch and binding of ligands such as angiotensin II (AT-II), endothelin I (ET-I) and phenylephrine (PE) to G-protein coupled receptors (13, 25). Subsequently, p38 can induce stress-mediated signaling by phosphorylating downstream transcription factors and cytoplasmic proteins. Selectively introducing MEK3 and MEK6, two upstream kinases required for p38 phosphorylation, into cardiomyocytes revealed that p38 MAPKβ can induce hypertrophy, while p38 MAPKα mediates apoptosis. Experiments in which ERK was specifically activated resulted in concentric hypertrophy without a reduction in myocardial function, and resistance to apoptotic cell death was increased (26). In contrast, selective activation of either JNK or p38 resulted in cardiomyopathy in transgenic animals (26, 27), highlighting the distinct roles of individual MAPK pathways.

Janus-associated kinase / signal transducers and activators of transcription. Janus-associated kinases (JAKs) are protein tyrosine kinases that are associated with cytokine receptors and become phosphorylated and activated upon ligands binding to the receptor. Signal transducers and activators of transcription (STATs) are latent transcription factors that can become activated by phosphorylation of a tyrosine residue through JAKs or other tyrosine kinase receptors or G-protein coupled receptors. The phosphorylated STATs

dimerize and translocate to the nucleus, where they bind to response elements and stimulate gene transcription (13). Activation of the JAK/STAT pathway occurs in pressure overloaded rat hearts (28, 29) and in stretched rat cardiomyocytes (30). The strain-induced activation of the JAK/STAT and ERK pathways may be mediated by the transmembrane glycoprotein gp130, an upstream activator of both the JAK/STAT and the ERK (31). Mice with a cardiac specific non-functional gp130 mutation have normal cardiac structure but show severe cardiac apoptosis accompanied by rapidly progressing dilated cardiomyopathy when subjected to pressure overload (32). Mechanical stress leads to the induction of gp130-dependent ligands such as cardiotrophin 1, suggesting that some of the activation may occur through paracrine signaling.

Protein kinase C. The protein kinase C (PKC) family of serine/threonine kinases consists of more than 10 different isoforms, each with distinct patterns of expression and agonist activation. Protein kinase C members act downstream of signal transduction pathways and can be activated by second messengers such as DAG or by phosphorylation through other kinases (33, 34). When activated, PKC proteins translocate to distinct subcellular sites and phosphorylate Raf directly or through Ras and thus activate the ERK pathway. In addition, PKC members can phosphorylate proteins such as IκB and thereby induce NF-κB translocation into the nucleus (12). The differential compartmental localization and translocation of the PKC isoforms is caused by each isoform binding to specific receptors for activated C kinases (RACKs) through unique binding domains (35-37). α–adrenergic receptor stimulation of cultured rat cardiac myocytes leads to translocation of PKCβ1 from the cytosol to the nucleus, PKCβII from fibrillar structures to the perinucleus and sarcolemma, PKCδ to the perinuclear region, and PKCε from the nucleus to the cytosol and fibrillar structures (35, 38). PKCα or PKCβ have been suspected to mediate phorbol 12-myristate 13-acetate (PMA) induced inhibition of calcium channels (39). Overexpression of PKCβ in the adult heart results in hypertrophy, fibrosis and systolic dysfunction (40, 41), possibly caused by PKCβ-mediated phosphorylation of troponin I (42). Recent studies suggest that PKCε, one of the more abundant isoforms in the adult heart, plays an important role in the normal trophic growth of cardiac myocytes during postnatal development (43, 44). Specific activation of PKCε

causes a protective effect on myocytes in isolated hearts subjected to global ischemia with reperfusion (43) and a physiological form of hypertrophy in transgenic mice (44), while inhibition of PKCε results in lethal heart failure from dilated cardiomyopathy. Furthermore, selective translocation of PKCε to particulate ventricular fractions has been observed during acute or chronic pressure overload (45), and Heidkamp et al.(46) demonstrated activation of focal adhesion kinase (FAK) by PKCε in neonatal rat ventricular myocytes. In cardiac myocytes, activation of PKCα and PKCβ has been shown to increase expression of genes encoding cardiac cytoskeletal proteins such as β-MHC, MLC-2, as well as ANF (47, 48).

Intracellular calcium signaling. Cardiomyocytes respond to stretch with an increase in intracellular calcium concentration and an increase in phospholipase C (PLC) activity. The stretch-induced increase in intracellular calcium is caused by an initial calcium influx through L-type calcium channels and stretch activated channels and a subsequent calcium induced release of calcium from the sarcoplasmic reticulum via the ryanodine receptor (49). Intracellular calcium can enhance calcium dependent protein kinase activities such as PKC and the calcium calmodulin dependent protein kinase, and the activity of the calcium/calmodulin dependent protein phosphatase calcineurin is mediated by intracellular calcium levels. Calmodulin acts as a calcium sensor and can trigger hypertrophic signals through interaction with the calcium/calmodulin dependent protein kinase (CaMK) and calcineurin. Overexpression of calmodulin leads to hypertrophy in *in vivo* models (50) and inhibiting calmodulin with antagonists inhibits hypertrophy of cultured cardiac myocytes (51). Transgenic mice overexpressing L-type voltage dependent calcium channels develop hypertrophy, ventricular fibrosis, and, over time, increased rates of apoptosis. Increased levels of PKCα activity were observed before the development of cardiac hypertrophy and failure, indicating the importance of closely controlled calcium homeostasis and PKC signaling in the development of cardiac hypertrophy (52). In addition, increased calcium levels can directly affect gene expression by influencing transcription, mRNA stability, and the translation of mRNA into protein (53, 54), even though intracellular and nuclear calcium are regulated independently (55, 56). In addition to the direct role of calcium channels, the Na+/H+

exchanger plays a role in cardiomyocyte stretch sensitivity (12). Satoh et al. (57) showed that chronic α-adrenergic stimulation results in decreased expression of sarcoplasmic reticulum calcium-ATPase and the calcium release channel while simultaneously increasing Na^{+}/Ca^{2+} exchanger expression, offering a possible explanation for the altered calcium dynamics observed in heart failure.

In addition to its role in intracellular signaling, intracellular calcium directly affects myocyte contractility, which in turn can trigger mechanotransduction pathways. Mutations in sarcomeric proteins can interrupt calcium balance by acting as ion traps for calcium in the defective sarcomere, therefore depleting the sarcoplasmic reticulum calcium stores (58). L-type channel inhibition may interrupt this process by attenuating the calcium-induced sarcoplasmic reticulum calcium release and therefore limiting sequestration of calcium in the mutant sarcomere (59). This model could also explain increased force generation of mutant muscle fibers at submaximal calcium levels (60, 61).

Calcineurin. Calcineurin is a serine-threonine phosphatase that is specifically activated by calcium/calmodulin. Its direct downstream target is nuclear factor of activated T cells (NFAT), but calcineurin can also activate JNK and certain PKC isoforms (62). Because of its calcium sensitivity, the calcineurin-dependent pathway links increases in intracellular calcium concentrations to cardiac hypertrophy. Inhibition of calcineurin prevents pressure overload-induced cardiac hypertrophy (63-65) and transgenic mice overexpressing activated forms of calcineurin develop cardiac hypertrophy, while treatment with cyclosporine, an calcineurin inhibitor, prevent this effect (13, 66). Furthermore, *in vitro* experiments with mechanically strained rat neonatal cardiomyocytes as well as an *in vivo* aortic constriction pressure-overload mouse model demonstrate that biomechanical strain directly induces gene expression for the calcineurin inhibitor Down Syndrome Critical Region-1 (DSCR1) in cardiac myocytes, indicating that mechanically induced DSCR1 may regulate the calcineurin-mediated hypertrophic response to mechanical overload (67).

Autocrine and paracrine factors. The heart consists of approximately 75% cardiac myocytes by volume, but myocytes contribute only 30-40% of the total cardiac cell number. Cardiac fibroblasts and endothelial cells account for the majority of the remaining 60-70% (68). Although they do not contribute to active contraction, cardiac fibroblasts play an important role in cardiac mechanotransduction. Strain application to cardiac fibroblasts causes activation of the ERK and JNK pathways and this induction is β1-integrin dependent (69, 70). Furthermore, strain-induced expression of collagen III and fibronectin mRNA as well as increased TGF-β1 activity has been observed in adult rat cardiac fibroblasts (71). Separate recent studies report a rapid increase in ERK and JNK activity in response to strain, as well as specific G-protein activation within 10-60 sec of applied strain (18, 69). The stretch induced response can be transmitted to cardiac myocytes by paracrine signaling. In addition, remodeling of the extracellular matrix (ECM) by cardiac fibroblasts can be interpreted as paracrine signaling that affects cardiac myocytes.

The release of autocrine and paracrine factors can thus amplify the initial growth signals triggered by mechanical stimulation of cardiac myocytes. Using conditioned medium from stretched cardiac myocytes and other cardiac cells, van Wamel et al. (72) determined that angiotensin II, endothelin-1 and TGF-β can act as autocrine and paracrine mediators of stretch induced cardiac hypertrophy. Interestingly, all these growth factors directly or indirectly activate members of the MAPK pathway, one of the major mechanosensitive signaling pathways. In a similar fashion, activation of the JAK/STAT pathway can be achieved by paracrine stimulation through cytokines such as cardiotrophin-1 or leukemia inhibitory factor (LIF).

1.1.3 Mechanosensors in cardiac myocytes

While strain induced signal transduction has been extensively studied *in vivo* and *in vitro*, the initial step required for signal transduction initiation, i.e. the cellular mechanosensor, has so far remained elusive. However, several potential candidates have been identified and are currently under investigation.

Stretch activated channels. The presence of stretch activated channels (SAC) has been established in several cell types, including cardiac myocytes. Membrane or cytoskeletal stress opens these cation channels, allowing Na^+ and Ca^{2+} to enter the cell. Stretch-induced increases in intracellular calcium occur in cultured heart cells, and this response is blocked by pre-incubation with the SAC blockers streptomycin and gadolinium (73-77). Direct calcium influx through stretch activated channels has been reported in mechanically stimulated chick cardiac myocytes (73). Cardiac myocytes isolated from several species respond to strain with membrane depolarization, prolongation of action potentials and extrasystoles (78, 79). Voltage clamp analysis reveals that stretch activated current through non-selective cation channels are the major cause of these strain-induced events. Interestingly, strain-induced currents were higher in hypertrophied hearts, giving a possible explanation for the increased occurrence of strain-induced arrhythmias in hypertrophic hearts. Strain-induced increases in spontaneous beating rate and a reduction in maximum diastolic and systolic potentials have also been observed in isolated rabbit sinoatrial node cells, consistent with the stretch activation of non-selective cation channels.

Integrins, extracellular matrix and focal adhesion kinase. Integrins are heterodimers composed of α and β subunits. Currently more than 24 different combinations of the at least 18 different α and 8 different β subunits are known, many expressed in a cell-type specific manner. The large extracellular domain binds to the extracellular matrix (ECM) or to ligands on other cells. The smaller cytoplasmic domains do not possess enzymatic activity but can bind to cytoskeletal molecules as well as to intracellular signaling molecules such as focal adhesion kinase (FAK), Rho or Rac. Integrins are capable of inside-out and outside-in signaling, i.e. ligand affinity and receptor clustering can be controlled by binding of intracellular signaling molecules (inside-out signaling), while receptor-ligand binding can modulate intracellular signaling in the cytoplasmic domain (outside-in signaling). The type of integrins expressed on a cell can vary in a spatial and temporal manner. Cardiac myocytes express integrin subunits α1, α3, α5, α6, α7, α9, α10, and β1 (80, 81). Expression of α1 and α5 subunits occurs in the embryonic heart, is downregulated postnatally and can be upregulated again following increased mechanical

loading. Increases in expression of α1, α5, α7B, β1A, and β1D integrins have been reported in mice and rat undergoing hypertrophy (80, 82, 83).

Recent studies show that cardiac myocytes transmit contractile forces to the extracellular matrix through attachment complexes comprised of vinculin, talin, integrin α6β1 and laminin (84, 85). These adhesion complexes are found at the costameres, the striated distributions of vinculin between Z-lines and the sarcolemma of cardiomyocytes. Integrins can function as mechanotransducers (86), either directly, e.g. by undergoing a conformational change that alters their binding characteristics, or indirectly, by the transmission of forces to other, cytoskeleton-associated proteins. Keller et al. (87) studied the role of integrin signaling by disrupting the integrin β1A function by replacing the extracellular domain with the Tac subunit of the IL-1 receptor coupled to the α-MHC promoter for cardiac specific expression in transgenic mice. Multiple transgenic founders with the highest expression levels died perinatally with severe replacement fibrosis. Surviving mice with high expression levels developed dilated cardiac hypertrophy, and even mice with moderate expression levels showed decreased baseline and mechanically-induced (aortic constriction) ERK1/2 and FAK signaling. Further evidence for the role of β1 integrins in α-adrenergic mediated cardiac hypertrophy has been demonstrated in neonatal rat cardiomyocytes, which undergo hypertrophy in response to phenylephrine (PE)(70). This response requires interaction with extracellular matrix proteins and is augmented by overexpression of β1 integrin. Suppression of integrin signaling by overexpression of free β1 cytoplasmic domain inhibited the hypertrophic response (70). Furthermore, Pham et al. (88) showed that the striated muscle specific β1D integrin and FAK participate in cardiac myocyte hypertrophy. Protein levels of β1D integrin increased by 350% in response to α-adrenergic stimulation and overexpression of β 1D integrin increases cell size and ANF synthesis. This hypertrophic response is inhibited by overexpression of free β1D cytoplasmic domains. In addition, FAK phosphorylation occurred rapidly and was sustained following α1-adrenergic stimulation, and FAK co-immunoprecipitated with β1D (88). The role of FAK in mediating ECM and integrin specific signaling has been further confirmed by studies with neonatal cardiomyocytes

cultured on lamin, fibronectin or non adhesive gelatin substrates. These cells demonstrated that adrenergic (PE) induced hypertrophy is ECM dependent (89) and that PE stimulation led to rapid phosphorylation of focal adhesion proteins including FAK and paxillin. Expression of a dominant negative mutant form of FAK (FRNK) attenuated PE stimulated hypertrophy. Rapid stretch induced activation of FAK has been demonstrated in neonatal rat cardiomyocytes, indicating that FAK can play an important role in mechanosensing and in the hypertrophic signal transduction pathway (90, 91).

Cytoskeleton/sarcomere. The sarcomere is not only the site of active force generation in cardiac muscle, it has also been proposed to act as a cellular mechanosensor. Calcium binding directly depends on sarcomere length, mostly due to length dependent binding of calcium to sarcomeric proteins, especially cardiac troponin C (92). More evidence for a role of sarcomeric calcium binding in mechanotransduction comes from experiments with mice with mutations in sarcomeric proteins. Mice with a missense mutation in the α-myosin heavy chain (Arg403Gln) gradually develop myocardial hypertrophy, myocyte disarray and fibrosis (93). Levels of calcium and the calcium binding protein calsequestrin in the sarcoplasmic reticulum are reduced even before the onset of morphological changes (94). Early administration of an L-type Ca2+ channel inhibitor restores normal levels of calcium and calsequestrin in the SR and prevents the development of hypertrophy in the Arg403Gln mouse, indicating that the disruption of sarcoplasmic reticulum calcium balance is an important contributor to cardiac hypertrophy. These findings confirm that mutant sarcomere proteins can cause abnormal calcium responses that initiate a hypertrophic response.

Another sarcomeric stretch sensor might be found in the Z-discs of the sarcomeres. The Z-disc complex serves as the anchoring site for titin, actin and nebulin filaments and crosslinks neighboring sarcomeres. Knöll et al.(95) showed that the Z-disc complex component muscle-specific LIM protein (MLP) interacts with telethonin (T-cap), a titin binding protein, and that the MLP/T-cap complex serves as a cardiac stretch sensor. MLP -/- cardiac myocytes have decreased passive mechanical stiffness and

impaired stretch-induced BNP expression. Furthermore, a human MLP mutation is associated with dilated cardiomyopathy.

In addition to sarcomeric proteins, other cytoskeletal components have been linked to mechanotransduction in cardiac myocytes. Schroder et al.(96) studied the role of microtubules in the adaptive response of developing chick myocardium to increased mechanical loading. Increased mechanical load resulted in increased levels of polymerized β-tubulin and an increase in microtubule density, associated with increased myocardial stiffness. Heling et al.(97) found that mRNA and protein expression of α- and β-tubulin, desmin, fibronectin, vimentin and vinculin were increased in biopsies taken from explanted failing human hearts compared with normal myocardium, suggesting cardiomyocyte remodeling and progressive fibrosis. However, it remains unclear whether these cytoskeletal proteins can act as primary sensors or if the observed changes merely reflect secondary effects. Sawada and Sheetz (98) found that in Triton X-100 treated mouse fibroblastic L-929 cells, more than 10 cytoplasmic proteins exhibited stretch-dependent binding to the cytoskeleton, including the focal contact proteins paxillin and focal adhesion kinase (FAK).

Nucleus. Dilated cardiomyopathies have been observed in humans and mouse models lacking functional nuclear envelope protein lamin A/C. Nuclei derived from lamin deficient cells have irregular structure indicating impaired nuclear mechanics. Impaired nuclear stability can lead to breakage of the nucleus resulting in direct cell death. Evidence of fragmented nuclei has been reported in skeletal muscle fibers form EDMD patients and in fibroblasts after nuclear isolation (99). In addition, recent studies report desmin intermediate filament-mediated changes in chromatin in response to mechanical strain and suggest that stretch-induced changes in chromatin can lead to activation of hypertrophy-associates genes (100). Mechanical connections between integrins, cytoskeletal filaments and the nucleus have also been demonstrated by micromanipulation with microbeads and micropipettes in endothelial cells (101). Mutations in nuclear envelope proteins such as lamin could interrupt some of these connections and impair nuclear mechanotransduction pathways.

1.1.4 Mechanotransduction in other cells types

Smooth muscle cells. Arteriolar smooth muscle cells respond to alterations in intraluminal pressure with changes in their myogenic tone, i.e. the level of partial contraction. This response appropriately adjusts the blood vessel diameter through vasoconstriction/vasodilatation, resulting in local blood flow autoregulation. Many of the immediate cellular responses are controlled through changes in intracellular calcium concentration. In addition to its direct role in muscle contractility, intracellular calcium is involved in the modulation of ion channel activity where it regulates the release of additional calcium from the sarcoplasmic reticulum or the extracellular space, in modifications in calcium sensitivity and in cell-to-cell communication (102). Mechanical strain also increases force production and calcium sensitivity in cultured airway smooth muscle cells (103). In addition to these short-term responses, mechanical strain activates several mechanosensitive genes in smooth muscle cells, many of them involved in cell signaling or in extracellular matrix degradation or remodeling. Cyclic strain induces rapid activation of platelet derived growth factor receptors in vascular smooth muscle cells followed by activation of ERK, JNK, and p38 MAPKs and AP-1 transcription factors (104). A number of other mechanosensitive genes in vascular smooth muscle cells have been identified using DNA microarray technology, including vascular endothelial growth factor (VEGF), Tenascin-C and plasminogen activator inhibitor (PAI)-1, whereas metalloproteinase-1 is downregulated in response to strain (105). Induction of Tenascin-C has also been demonstrated in cultured neonatal rat cardiac myocytes subjected to biaxial strain (106).

In addition to mechanosensing taking place in vascular smooth muscle cells, the myogenic response is further modulated by endothelial and neurohumoral factors released by other cell types (107).

Endothelial cells. Endothelial cells are exposed to fluid shear stress of $\approx$10 dyn/cm^2 (= 1 Pa or N/m^2), corresponding to 1 pN/μm^2. The glycocalix, a 150-400 nm thick fibrous meshwork on the luminal endothelial surface, has been proposed to function in part as a

mechanotransducer for this fluid shear stress. Deformations of the glyocalix through shear stress and drag forces in the endothelial surface layer can transmit and amplify mechanical forces to the endothelial cell membrane and to membrane and cytoskeletal proteins (108). Activation of these mechanosensors through changes in fluid shear stress levels trigger diverse signaling cascades that result in the activation of transcription factors and the interaction or generation of nitric oxide and reactive oxygen species than can serve as additional messengers (109). Endothelial cells subjected to fluid shear stress align in the direction of fluid flow, whereas cells exposed to cyclic strain orient themselves towards the direction of minimum substrate deformation, and the actin cytoskeleton remodels to form stress fibers in that direction (110). Cultured bovine aortic endothelial cells respond to shear stress with activation of focal adhesion kinase (FAK) and formation of FAK-Grb2-Sos complexes, leading to the activation of the mitogen-activated proteinkinases (MAPKs) ERK and JNK which induce the transcriptional activation of many immediate early genes. The shear stress activation of MAPKs depends on the integrity of the actin cytoskeleton and the upstream phospho-tyrosine kinase (PTK) signaling (111). Twisting integrin receptors with RGD-peptide coated beads leads to increased endothelin-1 expression in endothelial cells. This response can be abolished by blocking stretch-activated ion channels, chelating extracellular calcium with EGTA or reducing cytoskeletal tension with myosin inhibitors or disruption of the actin microfilaments (112). Receptors such as the vascular endothelial growth factor receptor on the luminal side or integrin receptors on the basal side have also been implicated as potential mechanosensors in endothelial cells. Additional candidates for mechanosensors include G-proteins, stretch-activated ion channels, intercellular junction proteins and lipid membrane fluidity (109, 113). Stretch-activated cation channels have been identified in human umbilical vein endothelial cells that respond to laminar shear stress leading to increased mechanosensitive calcium influx, serving as a potential mechanosensor in these cells (114). Patch-clamp experiments show that stretch-activated cation channel activity is persistently upregulated by fluid shear stress in endothelial cells, allowing mechanosensitive calcium entry into the cell (115). Heterotrimeric guanine nucleotide-binding proteins (G-proteins) reconstituted into phospholipids vesicles exposed to

physiological levels of shear stress exhibited a dose-dependent activity that was further modulated by membrane fluidity (113).

Chondrocytes and bone cells. Chondrocytes seeded in agarose gels exhibit increased intracellular calcium signaling in response to mechanical compression, and this response can be inhibited by blocking stretch-activated calcium channels or by chelating extracellular calcium (116). Furthermore, mechanical compression of cartilage leads to an increase in interstitial osmotic pressure, resulting in an additional decrease in cell volume and a transient increase in intracellular calcium concentrations, caused by mechanosensitive ion channels and an influx from extracellular calcium (117). Interestingly, changes in cell volume in response to physiological and pathological conditions have also been implicated in mechanosensitive signaling in hepatocytes that regulate proliterative signaling (118). The transcription factor c-fos is rapidly upregulated in response to mechanical load in bone cells (osteoblasts and osteocytes) as well as in cardiac cells and endothelial cells. Blocking stretch-activated calcium channels or chelating extracellular calcium inhibits this response (119). Osteoblasts probed with an atomic force microscope (AFM) show two distinct cellular responses, indicated by an increase in intracellular calcium. The first calcium transient, triggered through contact with the AFM probe, can be attributed to stretch-activated ion channels and is directly linked to membrane tension. The second calcium response is observed following stress relaxation and requires an intact microtubular cytoskeleton, confirming an involvement of the cytoskeleton in mechanotransduction (120).

1.2 Cellular mechanics

Many of the proposed mechanosensors described above require conformational changes in response to applied stress or strain, whether they are located in the cell membrane, the cytoskeleton or the nucleus. To better understand the sensitivity of these strain sensors, it is necessary to develop sophisticated methods that allow quantitative analysis of subcellular biomechanics to gauge local stress and strain levels in cells subjected to mechanical stimulation.

Current methods to study cellular biomechanics can be generally divided into three principles; (*i*) methods that deform portions of the cell and probe local mechanical properties, (*ii*) methods that stretch or deform the entire cell and measure global mechanical properties, or (*iii*) methods that mechanically stress or strain a large number (10^2-10^6) of cells simultaneously and yield estimates averaged over entire cell populations (121). Type (*i*) methods include microindentation, atomic force microscopy (AFM), magnetic twisting cytometry, magnetic bead microrheolgy and micropipette aspiration for small pipette diameters. These applications are discussed in more detail below. Stretching cells plated on elastic substrates or suspended between force transducers and micropipette aspiration with large pipette diameters are examples for type (*ii*) methods that lead to deformation of the entire cells. Type (*iii*) methods include fluid shear stress or strain stimulation of large cell populations plated in flow chambers or on mechanical strain devices. As these examples demonstrate, the distinction between the different types is not always quite distinct and some experimental procedures can be used for more than one type. In this work, we mainly focus on type (*i*) methods to explore subcellular mechanics in specific cellular domains, but also use type (*iii*) methods for gene-induction or protein-activation studies that require large quantities of mRNA or protein isolated from cells.

Some of the earliest experiments to measure cellular biomechanical properties are cell poking with microindenters and micropipette aspiration (122). In the micropipette aspiration experiments, a suspended cell is deformed by aspirating portions of the cell into a micropipette while precisely monitoring the suction pressure and monitoring the geometry changes, especially the radius of curvature, in the aspirated and the remaining cell portion. A variation of the micropipette set-up can be used for microindentation or micromanipulation experiments by using the microneedle to apply deformations to the cell surface or the cytoskeleton. In contrast to the aspiration experiments, the applied force can only be approximately measured based on the micropipette deflection.

Atomic force microscopy achieves sub-nanometer resolution in the vertical direction and nanometer resolution in the lateral directions. The applied force can be inferred from the measured cantilever deflection and is usually on the range of several pico- to nanonewton, depending on the available cantilever spring constant. However, since the probe tips serves as both the force transducer and the measurement probe, the displacement can only be measured on the cell surface at the site of the force application itself and measurements of displacements within the cytoskeleton are not possible. An additional application arises from using functionalized tips, i.e. coating the probe tip with specific antibodies or receptor or ligand proteins. In this set-up, the specific molecular interaction force can be measured, e.g. ≈30-120 pN for a single integrin/RGD-peptide bond (123).

Optical traps have a comparable resolution for positional measurements of spherical particles or cells organelles. In addition, optical traps allow precise manipulation of polystyrene beads on the cell surface or embedded in the cytoskeleton. The lateral bead position can be measured with sub-nanometer precision, but the applied forces are limited to below ≈100 pN as higher laser power can cause cellular damage. The most promising application therefore lies in molecular mechanics and single molecule manipulation.

Local viscoelastic properties can also be obtained by multiple particle tracking analysis of the Brownian displacement of small (≈0.1um) polystyrene particles imbedded in the cytoskeleton of fibroblasts by microinjection or endocytosis (124). However, laser tracking microrheology and diffusion particle tracking suffer the limitation that they can only measure elastic properties in very soft materials, since the diffusive force and Brownian movement are too small to induce detectable displacements in stiffer mediums. Their use is thus mainly limited to experiments with peptide solutions of low concentrations.

Magnetic twisting cytometry is based on the torque applied to ferromagnetic beads attached to the cells and has been applied to cardiomyocytes (125) and other cell

types (112, 126-128). Earlier adaptation of this method measured the induced bead rotation based on the magnetic field component induced by the ferromagnetic bead and suffered from the limitation that the measured rotation of the beads is averaged over a large sample of beads and cells, making it impossible to measure the mechanical response of individual cells. More recent studies analyze the bead displacement resulting from the bead rotation instead and can be used to study single cell mechanical properties. However, a study directly comparing results obtained from magnetic twisting cytometry and optical tweezers found that twisting cytometry yielded smaller apparent elasticity values (34-58 Pa vs. 29-259 Pa for adherent alveolar epithelial cells), and the degree of bead immersion into the cell had a large effect on the calculated results (129).

In contrast to the magnetic torque used in magnetic twisting cytometry, magnetic bead microrheology uses a magnetic field gradient generated by a single strong electromagnet to apply an attracting force to small paramagnetic beads. These magnetic tweezers or traps can apply a controlled force of up to 20 nN to beads bound to the cell surface or embedded in the cytoskeleton, and the induced bead displacements can be measured with ≈10 nm resolution using digital video analysis. Magnetic trap measurements have been used to demonstrate differences in integrin-receptor to cytoskeleton linkage in vinculin deficient cells (130) or to explore the difference between integrin and transferrin receptors coupling to the cytoskeleton in human aortic smooth muscle cells (131).

In addition to the methods that actively apply stress or strain to single cells, experimental procedures have been developed to measure forces generated within the cells and the mechanical interaction between adherent cells and their substrate. In the case of traction force microscopy, cells are plated on transparent polyacrylamide gels containing fluorescent polystyrene microspheres, and the cellular traction force can be deduced based on the displacement of these beads (132-134). An alternative method relies on microfabricated arrays of elastomeric microneedle-like posts that are deflected by the force exerted from the spreading cell (135). Additional methods have been developed specific for contractility measurements of striated muscle cells such as skeletal

muscle or adult cardiac myocytes. In the force clamp method, a force transducer and an actuator are attached to each end of the cell, allowing simultaneous measurement of force generated by the cell as well as total cell length (136). Sarcomere lengths can be measured simultaneously using laser diffraction (137) or video detection (138). Edge detection analysis is an alternative method to evaluate cardiac myocyte contractility. Isolated adult myocytes plated on glass coverslips are paced by electric field stimulation while cell dimensions are tracked using video edge detection to determine cell and sarcomere shortening percentage and velocity. Force clamp systems have the advantage that sarcomere resting length can be adjusted to physiological levels and that the force generated by the cardiomyocytes can be quantified. However, the measurement only reflects the total force integrated over the entire cell length and cross-section and does not account for local differences. The edge detection method on the other hand does not allow to directly measure the generated force or to apply controlled stresses or strains to the cardiomyocytes. The only resistance to cell contraction is the stiff substrate the cells are attached to. Furthermore, the lack of pre-stretch leaves the cell in a shortened stage with decreased sarcomere length.

Fluorescence labeling and advanced imaging techniques can be combined with the above stress or strain application methods to obtain intracellular displacement maps. Using GFP-labeled cytoskeletal protein, cytoskeletal deformations in rat embryo fibroblasts have been visualized while manipulating the cells with laminin-coated or uncoated glass microneedles. These experiments reveal an apparent three layer model of cellular mechanics, comprised of an elastic nucleus surrounded by viscoelastic cytoplasmic microtubules enclosed by an elastic shell under sustained tension (139). Spatiotemporal strain maps in live endothelial cells have also been obtained using GFP-labeled vimentin before and after onset of shear stress (140), and similar strain maps have been generated using fluorescent protein-labeled mitochondria or cytoskeletal components in human airway smooth muscle cells in response to a force applied through a RGD-coated magnetic bead on the apical cell surface (141). The intrinsic striation pattern found in adult cardiac myocytes imaged under bright field can also be utilized to achieve high resolution strain maps in response to applied forces through magnetic beads

using cross-correlation algorithm based digital image analysis (142). Furthermore, three dimensional cellular displacement maps can be achieved by combining a three dimensional magnetic manipulator with two-photon imaging to track small fluorescent beads endocytosed by aortic smooth muscle cells (143). By combining single cell force application with traction force spectroscopy, it is also possible to study force transmission through the cell to the underlying substrate

In addition to these exclusively mechanically measurements, the above force or strain application methods can be combined with emerging imaging techniques for single cell signaling to study mechanotransduction events in live cells, e.g. increases in local calcium concentration after force application with an AFM probe (120) or formation of focal adhesions on forced microbeads adherent to the cell surface (144). These measurements can then provide important new information on the spatio-temporal distribution and dynamics of mechanotransduction.

1.3 Thesis objective

The research conducted in this thesis will serve to develop quantitative methods to study subcellular biomechanics and mechanotransduction. Subcellular mechanics refer to the specific functional and spatial cellular domains such as the cell membrane, the cytoskeleton and the nucleus. Externally applied forces and deformations are transmitted through the extracellular matrix to cellular focal adhesion sites, the cellular contact points with the extracellular matrix. Cell surface receptors such as integrins and integrin associated proteins (e.g. tetraspanin CD151) transmit applied forces across the cell membrane. Integrins bind to talin and several other proteins to form the focal adhesion complex that physically connects the extracellular matrix to the cytoskeleton. The cytoskeleton consists of a network of the actin filaments, the intermediate filament network and the microtubules that emerge from the nucleus and spread through the cells. All three components of the cytoskeleton are thought to connect to the nucleus through the nuclear envelope, and are thus capable of transmitting forces from the outside of the cell to the nuclear envelope and the chromatin, that is physically connected to the nuclear

envelope and intranuclear structures of lamins and actin. In addition to this outside-in force transmission, the cytoskeleton can itself generate forces through actin-myosin interaction or protein polymerization dynamics, and these forces are transmitted to the extracellular matrix. Furthermore, cells are capable of undergoing significant changes and remodeling in the focal adhesions, cytoskeletal architecture and nuclear shape and position in response to mechanical stimulation. Consequently, it is important to develop tools to independently study subcellular biomechanics in each cellular domain, i.e. the coupling between the cytoskeleton and the extracellular matrix, the cytoskeletal biomechanics that dominate the overall cellular mechanical response and the nuclear and nuclear envelope mechanics that can affect cellular viability and mechanosensing.

***Specific Aim #1*:** Develop an experimental system to apply controlled forces to single cells to quantitatively measure induced cellular deformations. This includes design and set-up of a single pole magnetic trap and control software, digital imaging equipment and software development of image analysis algorithms to obtain high resolution displacement and deformation results.

***Specific Aim #2*:** Apply the experimental system to quantitatively analyze the effect of specific mutations on local subcellular biomechanical properties in the following domains: (*i*) mechanical coupling between cytoskeleton and extracellular matrix, (*ii*) cytoskeletal mechanics, and (*iii*) nuclear mechanics

***Specific Aim #3*:** Study the effects of specific mutations and altered mechanical properties on mechanotransduction. This includes the analysis of mechanosensitive gene induction in response to strain or stress application, activation of mechanosensitive signaling pathways such as NF-κB, and the overall sensitivity (necrosis/apoptosis) of cells to mechanical stimulation.

Chapter 2 describes the detailed design and calibration steps for our magnetic trap system and a cellular strain device, as well as the image analysis algorithms developed to interpret experimental data obtained from these systems. The first application of the

newly developed magnetic trap system is described in Chapter 3, in which we study the effects of a C-terminal modification of the tetraspanin CD151 on cellular adhesion forces and adhesion strengthening. Subsequently, our magnetic trap system is used to validate a viscoelastic finite element model of cytoskeletal mechanics developed by Helene Karcher, and these experiments are described in detail in Chapter 4. In work described in Chapter 5, magnetic bead microrheology is applied to study the effect of cardiomyopathy causing mutations on the mechanical properties of adult mouse cardiac myocytes. Newly developed image analysis software is used to obtain deformation maps in these cardiac myocytes, and single sarcomere tracking algorithms reveal the quick decay of induced deformations for forces applied in the sarcomere direction. Finally, Chapter 6 presents new findings on the role of the nuclear envelope proteins lamin A/C on nuclear mechanics and its effects on mechanotransduction. These data reveal interesting connections between nuclear and cytoskeletal mechanics, indicating that there is a tight coupling between the intracellular mechanical domains.

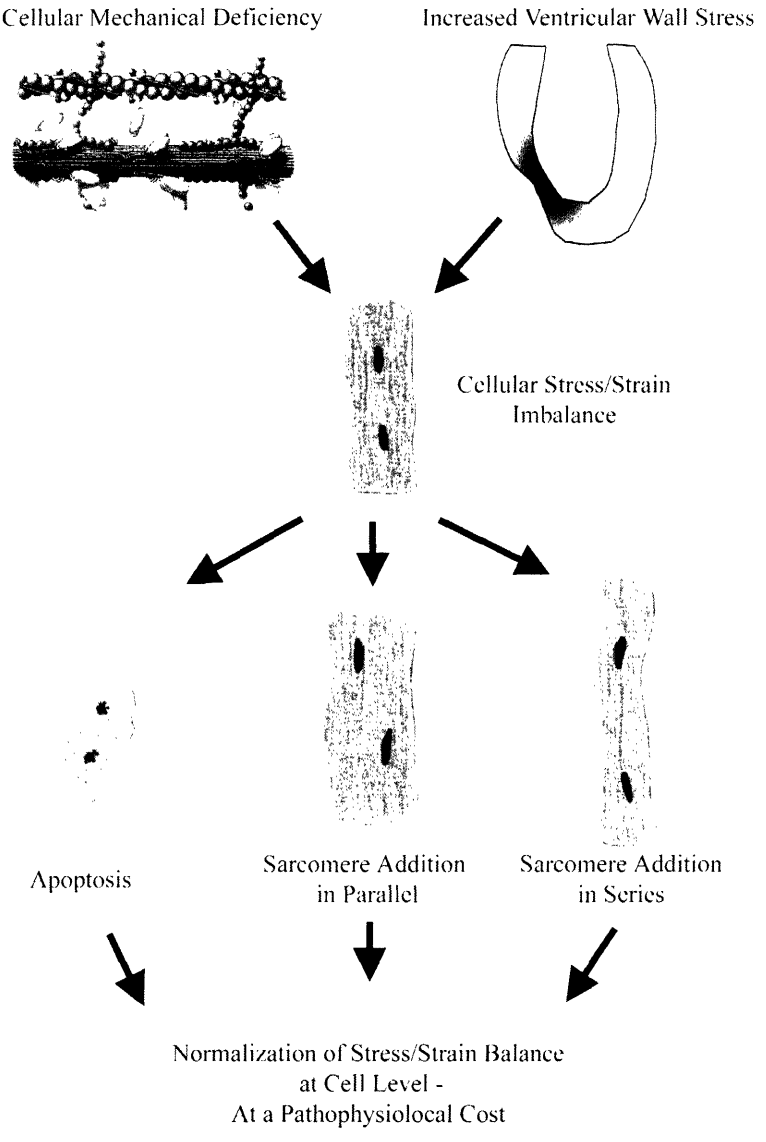


Figure 1.1. Role of mechanotransduction in cardiac hypertrophy. Changes in ventricular mechanical loading, localized tissue damage following infarction or by changes in cellular contractility caused by congenital mutations can result in an imbalance of the cellular stress/strain level. Initiated by mechanotransduction pathways, cells respond with apoptosis or hypertrophy, cellular growth by the addition of sarcomeres in parallel or in series. This response leads to a normalization of the cellular stress/strain balance, but is eventually associated with pathophysiological changes such as re-induction of fetal genes and further changes in myocyte contractility.

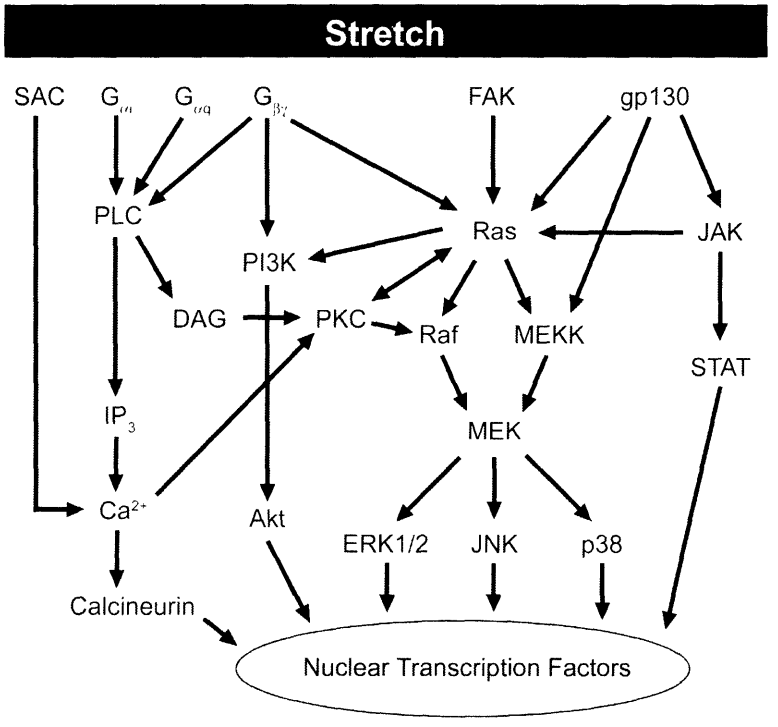


Figure 1.2. Signal transduction pathways involved in the cellular response to mechanical stimulation. Mechanical stretch can lead to activation of stretch-activated channels, G-protein coupled proteins, focal adhesion kinase and the glycoprotein gp130. Activation of these proteins can initiate several downstream signaling pathways such as MAPK, JAK/STAT, PKC and PI3K, leading to changes in intracellular calcium concentration and the activation of several nuclear transcription factors such as c-fos, c-jun, and c-myc. See text for more details. Abbreviations: SAC, stretch-activated channel; FAK, focal adhesion kinase; PLC, phospholipase C; IP_3, inositol-triphosphate; DAG, diacylglycerol; PI3K, phosphatidylinositol 3-kinase; PKC, protein kinase C; MEKK, mitogen-activated protein kinase kinase kinase; MEK, mitogen-activated protein kinase kinase; ERK, extracellular signal-regulated kinase; JNK, c-Jun N-terminal kinase; p38, p38 mitogen-activated protein kinase; JAK, Janus-associated kinase; STAT, signal transducers and activators of transcription

1.4 References

1. Hamill, O. P. & Martinac, B. (2001) *Physiol Rev* **81,** 685-740.
2. Gillespie, P. G. & Walker, R. G. (2001) *Nature* **413,** 194-202.
3. Davies, P. F. (1995) *Physiol Rev* **75,** 519-60.
4. Grossman, W., Jones, D. & McLaurin, L. P. (1975) *J Clin Invest* **56,** 56-64.
5. Sadoshima, J. & Izumo, S. (1997) *Annu Rev Physiol* **59,** 551-71.
6. Marian, A. J. (2000) *Lancet* **355,** 58-60.
7. Dipla, K., Mattiello, J. A., Jeevanandam, V., Houser, S. R. & Margulies, K. B. (1998) *Circulation* **97,** 2316-22.
8. Ritter, M., Su, Z., Xu, S., Shelby, J. & Barry, W. H. (2000) *J Mol Cell Cardiol* **32,** 577-84.
9. de Jonge, N., van Wichen, D. F., Schipper, M. E., Lahpor, J. R., Gmelig-Meyling, F. H., Robles de Medina, E. O. & de Weger, R. A. (2002) *J Am Coll Cardiol* **39,** 963-9.
10. Hunter, J. J. & Chien, K. R. (1999) *N Engl J Med* **341,** 1276-83.
11. Tavi, P., Laine, M., Weckstrom, M. & Ruskoaho, H. (2001) *Trends Pharmacol Sci* **22,** 254-60.
12. Ruwhof, C. & van der Laarse, A. (2000) *Cardiovasc Res* **47,** 23-37.
13. Molkentin, J. D. & Dorn, I. G., 2nd (2001) *Annu Rev Physiol* **63,** 391-426.
14. Sadoshima, J. & Izumo, S. (1993) *Embo J* **12,** 1681-92.
15. Crespo, P., Xu, N., Simonds, W. F. & Gutkind, J. S. (1994) *Nature* **369,** 418-20.
16. Pumiglia, K. M., LeVine, H., Haske, T., Habib, T., Jove, R. & Decker, S. J. (1995) *J Biol Chem* **270,** 14251-4.
17. Naga Prasad, S. V., Esposito, G., Mao, L., Koch, W. J. & Rockman, H. A. (2000) *J Biol Chem* **275,** 4693-8.
18. Gudi, S. R., Lee, A. A., Clark, C. B. & Frangos, J. A. (1998) *Am J Physiol* **274,** C1424-8.
19. Adams, J. W., Sakata, Y., Davis, M. G., Sah, V. P., Wang, Y., Liggett, S. B., Chien, K. R., Brown, J. H. & Dorn, G. W., 2nd (1998) *Proc Natl Acad Sci U S A* **95,** 10140-5.
20. Esposito, G., Rapacciuolo, A., Naga Prasad, S. V., Takaoka, H., Thomas, S. A., Koch, W. J. & Rockman, H. A. (2002) *Circulation* **105,** 85-92.
21. Sano, M. & Schneider, M. D. (2002) *Circulation* **105,** 8-10.
22. Yazaki, Y. & Komuro, I. (1992) *Basic Res Cardiol* **87 Suppl 2,** 11-8.
23. Komuro, I., Kudo, S., Yamazaki, T., Zou, Y., Shiojima, I. & Yazaki, Y. (1996) *Faseb J* **10,** 631-6.
24. Choukroun, G., Hajjar, R., Fry, S., del Monte, F., Haq, S., Guerrero, J. L., Picard, M., Rosenzweig, A. & Force, T. (1999) *J Clin Invest* **104,** 391-8.
25. Seko, Y., Takahashi, N., Tobe, K., Kadowaki, T. & Yazaki, Y. (1999) *Biochem Biophys Res Commun* **259,** 8-14.
26. Bueno, O. F., De Windt, L. J., Tymitz, K. M., Witt, S. A., Kimball, T. R., Klevitsky, R., Hewett, T. E., Jones, S. P., Lefer, D. J., Peng, C. F., Kitsis, R. N. & Molkentin, J. D. (2000) *Embo J* **19,** 6341-50.
27. Liao, P., Georgakopoulos, D., Kovacs, A., Zheng, M., Lerner, D., Pu, H., Saffitz, J., Chien, K., Xiao, R. P., Kass, D. A. & Wang, Y. (2001) *Proc Natl Acad Sci U S A* **98,** 12283-8.

28. Pan, J., Fukuda, K., Kodama, H., Makino, S., Takahashi, T., Sano, M., Hori, S. & Ogawa, S. (1997) *Circ Res* **81,** 611-7.
29. Pan, J., Fukuda, K., Kodama, H., Sano, M., Takahashi, T., Makino, S., Kato, T., Manabe, T., Hori, S. & Ogawa, S. (1998) *Heart Vessels* **13,** 199-208.
30. Pan, J., Fukuda, K., Saito, M., Matsuzaki, J., Kodama, H., Sano, M., Takahashi, T., Kato, T. & Ogawa, S. (1999) *Circ Res* **84,** 1127-36.
31. Kunisada, K., Hirota, H., Fujio, Y., Matsui, H., Tani, Y., Yamauchi-Takihara, K. & Kishimoto, T. (1996) *Circulation* **94,** 2626-32.
32. Hirota, H., Chen, J., Betz, U. A., Rajewsky, K., Gu, Y., Ross, J., Jr., Muller, W. & Chien, K. R. (1999) *Cell* **97,** 189-98.
33. Nishizuka, Y. (1986) *Science* **233,** 305-12.
34. Nishizuka, Y. & Kikkawa, U. (2003) *Methods Mol Biol* **233,** 9-18.
35. Disatnik, M. H., Buraggi, G. & Mochly-Rosen, D. (1994) *Exp Cell Res* **210,** 287-97.
36. Mochly-Rosen, D. (1995) *Science* **268,** 247-51.
37. Mackay, K. & Mochly-Rosen, D. (2001) *J Mol Cell Cardiol* **33,** 1301-7.
38. Disatnik, M. H., Jones, S. N. & Mochly-Rosen, D. (1995) *J Mol Cell Cardiol* **27,** 2473-81.
39. Zhang, Z. H., Johnson, J. A., Chen, L., El-Sherif, N., Mochly-Rosen, D. & Boutjdir, M. (1997) *Circ Res* **80,** 720-9.
40. Wakasaki, H., Koya, D., Schoen, F. J., Jirousek, M. R., Ways, D. K., Hoit, B. D., Walsh, R. A. & King, G. L. (1997) *Proc Natl Acad Sci U S A* **94,** 9320-5.
41. Bowman, J. C., Steinberg, S. F., Jiang, T., Geenen, D. L., Fishman, G. I. & Buttrick, P. M. (1997) *J Clin Invest* **100,** 2189-95.
42. Takeishi, Y. & Walsh, R. A. (2001) *Acta Physiol Scand* **173,** 103-11.
43. Dorn, G. W., 2nd, Souroujon, M. C., Liron, T., Chen, C. H., Gray, M. O., Zhou, H. Z., Csukai, M., Wu, G., Lorenz, J. N. & Mochly-Rosen, D. (1999) *Proc Natl Acad Sci U S A* **96,** 12798-803.
44. Mochly-Rosen, D., Wu, G., Hahn, H., Osinska, H., Liron, T., Lorenz, J. N., Yatani, A., Robbins, J. & Dorn, G. W., 2nd (2000) *Circ Res* **86,** 1173-9.
45. Schunkert, H., Sadoshima, J., Cornelius, T., Kagaya, Y., Weinberg, E. O., Izumo, S., Riegger, G. & Lorell, B. H. (1995) *Circ Res* **76,** 489-97.
46. Heidkamp, M. C., Bayer, A. L., Scully, B. T., Eble, D. M. & Samarel, A. M. (2003) *Am J Physiol Heart Circ Physiol* **285,** H1684-96.
47. Kariya, K., Karns, L. R. & Simpson, P. C. (1991) *J Biol Chem* **266,** 10023-6.
48. Shubeita, H. E., Martinson, E. A., Van Bilsen, M., Chien, K. R. & Brown, J. H. (1992) *Proc Natl Acad Sci U S A* **89,** 1305-9.
49. Ruwhof, C., van Wamel, J. T., Noordzij, L. A., Aydin, S., Harper, J. C. & van der Laarse, A. (2001) *Cell Calcium* **29,** 73-83.
50. Gruver, C. L., DeMayo, F., Goldstein, M. A. & Means, A. R. (1993) *Endocrinology* **133,** 376-88.
51. McDonough, P. M. & Glembotski, C. C. (1992) *J Biol Chem* **267,** 11665-8.
52. Muth, J. N., Bodi, I., Lewis, W., Varadi, G. & Schwartz, A. (2001) *Circulation* **103,** 140-7.
53. Rosen, L. B., Ginty, D. D. & Greenberg, M. E. (1995) *Adv Second Messenger Phosphoprotein Res* **30,** 225-53.

54. Palfrey, H. C. & Nairn, A. C. (1995) *Adv Second Messenger Phosphoprotein Res* **30,** 191-223.
55. Echevarria, W., Leite, M. F., Guerra, M. T., Zipfel, W. R. & Nathanson, M. H. (2003) *Nat Cell Biol* **5,** 440-6.
56. Itano, N., Okamoto, S., Zhang, D., Lipton, S. A. & Ruoslahti, E. (2003) *Proc Natl Acad Sci U S A* **100,** 5181-6.
57. Satoh, N., Suter, T. M., Liao, R. & Colucci, W. S. (2000) *Circulation* **102,** 2249-54.
58. Seidman, J. G. & Seidman, C. (2001) *Cell* **104,** 557-67.
59. Fatkin, D., McConnell, B. K., Mudd, J. O., Semsarian, C., Moskowitz, I. G., Schoen, F. J., Giewat, M., Seidman, C. E. & Seidman, J. G. (2000) *J Clin Invest* **106,** 1351-9.
60. Blanchard, E., Seidman, C., Seidman, J. G., LeWinter, M. & Maughan, D. (1999) *Circ Res* **84,** 475-83.
61. Tyska, M. J., Hayes, E., Giewat, M., Seidman, C. E., Seidman, J. G. & Warshaw, D. M. (2000) *Circ Res* **86,** 737-44.
62. De Windt, L. J., Lim, H. W., Haq, S., Force, T. & Molkentin, J. D. (2000) *J Biol Chem* **275,** 13571-9.
63. Sussman, M. A., Lim, H. W., Gude, N., Taigen, T., Olson, E. N., Robbins, J., Colbert, M. C., Gualberto, A., Wieczorek, D. F. & Molkentin, J. D. (1998) *Science* **281,** 1690-3.
64. Shimoyama, M., Hayashi, D., Takimoto, E., Zou, Y., Oka, T., Uozumi, H., Kudoh, S., Shibasaki, F., Yazaki, Y., Nagai, R. & Komuro, I. (1999) *Circulation* **100,** 2449-54.
65. Wilkins, B. J. & Molkentin, J. D. (2002) *J Physiol* **541,** 1-8.
66. Molkentin, J. D., Lu, J. R., Antos, C. L., Markham, B., Richardson, J., Robbins, J., Grant, S. R. & Olson, E. N. (1998) *Cell* **93,** 215-28.
67. Wang, Y., De Keulenaer, G. W., Weinberg, E. O., Muangman, S., Gualberto, A., Landschulz, K. T., Turi, T. G., Thompson, J. F. & Lee, R. T. (2002) *Am J Physiol Heart Circ Physiol* **283,** H533-9.
68. Brutsaert, D. L. (2003) *Physiol Rev* **83,** 59-115.
69. MacKenna, D. A., Dolfi, F., Vuori, K. & Ruoslahti, E. (1998) *J Clin Invest* **101,** 301-10.
70. Ross, R. S., Pham, C., Shai, S. Y., Goldhaber, J. I., Fenczik, C., Glembotski, C. C., Ginsberg, M. H. & Loftus, J. C. (1998) *Circ Res* **82,** 1160-72.
71. Lee, A. A., Delhaas, T., McCulloch, A. D. & Villarreal, F. J. (1999) *J Mol Cell Cardiol* **31,** 1833-43.
72. van Wamel, A. J., Ruwhof, C., van der Valk-Kokshoom, L. E., Schrier, P. I. & van der Laarse, A. (2001) *Mol Cell Biochem* **218,** 113-24.
73. Sigurdson, W., Ruknudin, A. & Sachs, F. (1992) *Am J Physiol* **262,** H1110-5.
74. Gannier, F., White, E., Lacampagne, A., Garnier, D. & Le Guennec, J. Y. (1994) *Cardiovasc Res* **28,** 1193-8.
75. Gannier, F., White, E., Garnier & Le Guennec, J. Y. (1996) *Cardiovasc Res* **32,** 158-67.
76. Tatsukawa, Y., Kiyosue, T. & Arita, M. (1997) *Heart Vessels* **12,** 128-35.
77. Belus, A. & White, E. (2003) *J Physiol* **546,** 501-9.

78. Kamkin, A., Kiseleva, I. & Isenberg, G. (2000) *Cardiovasc Res* **48,** 409-20.
79. Kamkin, A., Kiseleva, I. & Isenberg, G. (2003) *Cardiovasc Res* **57,** 793-803.
80. Ross, R. S. & Borg, T. K. (2001) *Circ Res* **88,** 1112-9.
81. Collo, G., Starr, L. & Quaranta, V. (1993) *J Biol Chem* **268,** 19019-24.
82. Terracio, L., Rubin, K., Gullberg, D., Balog, E., Carver, W., Jyring, R. & Borg, T. K. (1991) *Circ Res* **68,** 734-44.
83. Babbitt, C. J., Shai, S. Y., Harpf, A. E., Pham, C. G. & Ross, R. S. (2002) *Histochem Cell Biol* **118,** 431-9.
84. Imanaka-Yoshida, K., Enomoto-Iwamoto, M., Yoshida, T. & Sakakura, T. (1999) *Cell Motil Cytoskeleton* **42,** 1-11.
85. Danowski, B. A., Imanaka-Yoshida, K., Sanger, J. M. & Sanger, J. W. (1992) *J Cell Biol* **118,** 1411-20.
86. Ingber, D. (1991) *Curr Opin Cell Biol* **3,** 841-8.
87. Keller, R. S., Shai, S. Y., Babbitt, C. J., Pham, C. G., Solaro, R. J., Valencik, M. L., Loftus, J. C. & Ross, R. S. (2001) *Am J Pathol* **158,** 1079-90.
88. Pham, C. G., Harpf, A. E., Keller, R. S., Vu, H. T., Shai, S. Y., Loftus, J. C. & Ross, R. S. (2000) *Am J Physiol Heart Circ Physiol* **279,** H2916-26.
89. Taylor, J. M., Rovin, J. D. & Parsons, J. T. (2000) *J Biol Chem* **275,** 19250-7.
90. Torsoni, A. S., Constancio, S. S., Nadruz, W., Jr., Hanks, S. K. & Franchini, K. G. (2003) *Circ Res* **93,** 140-7.
91. Kuppuswamy, D. (2002) *Circ Res* **90,** 1240-2.
92. Calaghan, S. C. & White, E. (1999) *Prog Biophys Mol Biol* **71,** 59-90.
93. Kim, S. J., Iizuka, K., Kelly, R. A., Geng, Y. J., Bishop, S. P., Yang, G., Kudej, A., McConnell, B. K., Seidman, C. E., Seidman, J. G. & Vatner, S. F. (1999) *Am J Physiol* **276,** H1780-7.
94. Semsarian, C., Ahmad, I., Giewat, M., Georgakopoulos, D., Schmitt, J. P., McConnell, B. K., Reiken, S., Mende, U., Marks, A. R., Kass, D. A., Seidman, C. E. & Seidman, J. G. (2002) *J Clin Invest* **109,** 1013-20.
95. Knoll, R., Hoshijima, M., Hoffman, H. M., Person, V., Lorenzen-Schmidt, I., Bang, M. L., Hayashi, T., Shiga, N., Yasukawa, H., Schaper, W., McKenna, W., Yokoyama, M., Schork, N. J., Omens, J. H., McCulloch, A. D., Kimura, A., Gregorio, C. C., Poller, W., Schaper, J., Schultheiss, H. P. & Chien, K. R. (2002) *Cell* **111,** 943-55.
96. Schroder, E. A., Tobita, K., Tinney, J. P., Foldes, J. K. & Keller, B. B. (2002) *Circ Res* **91,** 353-9.
97. Heling, A., Zimmermann, R., Kostin, S., Maeno, Y., Hein, S., Devaux, B., Bauer, E., Klovekorn, W. P., Schlepper, M., Schaper, W. & Schaper, J. (2000) *Circ Res* **86,** 846-53.
98. Sawada, Y. & Sheetz, M. P. (2002) *J Cell Biol* **156,** 609-15.
99. Markiewicz, E., Venables, R., Mauricio Alvarez, R., Quinlan, R., Dorobek, M., Hausmanowa-Petrucewicz, I. & Hutchison, C. (2002) *J Struct Biol* **140,** 241-53.
100. Bloom, S., Lockard, V. G. & Bloom, M. (1996) *J Mol Cell Cardiol* **28,** 2123-7.
101. Maniotis, A. J., Chen, C. S. & Ingber, D. E. (1997) *Proc Natl Acad Sci U S A* **94,** 849-54.
102. Hill, M. A., Zou, H., Potocnik, S. J., Meininger, G. A. & Davis, M. J. (2001) *J Appl Physiol* **91,** 973-83.

103. Smith, P. G., Roy, C., Fisher, S., Huang, Q. Q. & Brozovich, F. (2000) *J Appl Physiol* **89,** 2092-8.
104. Hu, Y., Bock, G., Wick, G. & Xu, Q. (1998) *Faseb J* **12,** 1135-42.
105. Feng, Y., Yang, J. H., Huang, H., Kennedy, S. P., Turi, T. G., Thompson, J. F., Libby, P. & Lee, R. T. (1999) *Circ Res* **85,** 1118-23.
106. Yamamoto, K., Dang, Q. N., Kennedy, S. P., Osathanondh, R., Kelly, R. A. & Lee, R. T. (1999) *J Biol Chem* **274,** 21840-6.
107. Davis, M. J. & Hill, M. A. (1999) *Physiol Rev* **79,** 387-423.
108. Weinbaum, S., Zhang, X., Han, Y., Vink, H. & Cowin, S. C. (2003) *Proc Natl Acad Sci U S A* **100,** 7988-95.
109. Fisher, A. B., Chien, S., Barakat, A. I. & Nerem, R. M. (2001) *Am J Physiol Lung Cell Mol Physiol* **281,** L529-33.
110. Wang, J. H., Goldschmidt-Clermont, P., Wille, J. & Yin, F. C. (2001) *J Biomech* **34,** 1563-72.
111. Li, S., Kim, M., Hu, Y. L., Jalali, S., Schlaepfer, D. D., Hunter, T., Chien, S. & Shyy, J. Y. (1997) *J Biol Chem* **272,** 30455-62.
112. Chen, J., Fabry, B., Schiffrin, E. L. & Wang, N. (2001) *Am J Physiol Cell Physiol* **280,** C1475-84.
113. Gudi, S., Nolan, J. P. & Frangos, J. A. (1998) *Proc Natl Acad Sci U S A* **95,** 2515-9.
114. Brakemeier, S., Eichler, I., Hopp, H., Kohler, R. & Hoyer, J. (2002) *Cardiovasc Res* **53,** 209-18.
115. Brakemeier, S., Kersten, A., Eichler, I., Grgic, I., Zakrzewicz, A., Hopp, H., Kohler, R. & Hoyer, J. (2003) *Cardiovasc Res* **60,** 488-96.
116. Roberts, S. R., Knight, M. M., Lee, D. A. & Bader, D. L. (2001) *J Appl Physiol* **90,** 1385-91.
117. Erickson, G. R., Alexopoulos, L. G. & Guilak, F. (2001) *J Biomech* **34,** 1527-35.
118. Kim, R. D., Stein, G. S. & Chari, R. S. (2001) *J Cell Biochem* **83,** 56-69.
119. Peake, M. A., Cooling, L. M., Magnay, J. L., Thomas, P. B. & El Haj, A. J. (2000) *J Appl Physiol* **89,** 2498-507.
120. Charras, G. T. & Horton, M. A. (2002) *Biophys J* **82,** 2970-81.
121. Bao, G. & Suresh, S. (2003) *Nat Mater* **2,** 715-25.
122. Hochmuth, R. M. (2000) *J Biomech* **33,** 15-22.
123. Lehenkari, P. P. & Horton, M. A. (1999) *Biochem Biophys Res Commun* **259,** 645-50.
124. Tseng, Y., Kole, T. P. & Wirtz, D. (2002) *Biophys J* **83,** 3162-76.
125. Tagawa, H., Wang, N., Narishige, T., Ingber, D. E., Zile, M. R. & Cooper, G. t. (1997) *Circ Res* **80,** 281-9.
126. Wang, N. & Ingber, D. E. (1995) *Biochem Cell Biol* **73,** 327-35.
127. Wang, N. & Stamenovic, D. (2000) *Am J Physiol Cell Physiol* **279,** C188-94.
128. Berrios, J. C., Schroeder, M. A. & Hubmayr, R. D. (2001) *J Appl Physiol* **91,** 65-73.
129. Laurent, V. M., Henon, S., Planus, E., Fodil, R., Balland, M., Isabey, D. & Gallet, F. (2002) *J Biomech Eng* **124,** 408-21.
130. Alenghat, F. J., Fabry, B., Tsai, K. Y., Goldmann, W. H. & Ingber, D. E. (2000) *Biochem Biophys Res Commun* **277,** 93-9.

131. Huang, H., Kamm, R. D., So, P. T. & Lee, R. T. (2001) *Hypertension* **38,** 1158-61.
132. Dembo, M. & Wang, Y. L. (1999) *Biophys J* **76,** 2307-16.
133. Oliver, T., Dembo, M. & Jacobson, K. (1999) *J Cell Biol* **145,** 589-604.
134. Marganski, W. A., De Biase, V. M., Burgess, M. L. & Dembo, M. (2003) *Cardiovasc Res* **60,** 547-56.
135. Tan, J. L., Tien, J., Pirone, D. M., Gray, D. S., Bhadriraju, K. & Chen, C. S. (2003) *Proc Natl Acad Sci U S A* **100,** 1484-9.
136. Tasche, C., Meyhofer, E. & Brenner, B. (1999) *Am J Physiol* **277,** H2400-8.
137. Iwazumi, T. & Pollack, G. H. (1979) *IEEE Trans Biomed Eng* **26,** 86-93.
138. Roos, K. P. & Brady, A. J. (1982) *Biophys J* **40,** 233-44.
139. Heidemann, S. R., Kaech, S., Buxbaum, R. E. & Matus, A. (1999) *J Cell Biol* **145,** 109-22.
140. Helmke, B. P., Rosen, A. B. & Davies, P. F. (2003) *Biophys J* **84,** 2691-9.
141. Hu, S., Chen, J., Fabry, B., Numaguchi, Y., Gouldstone, A., Ingber, D. E., Fredberg, J. J., Butler, J. P. & Wang, N. (2003) *Am J Physiol Cell Physiol* **285,** C1082-90.
142. Lammerding, J., Huang, H., So, P. T., Kamm, R. D. & Lee, R. T. (2003) *IEEE Eng Med Biol Mag* **22,** 124-7.
143. Huang, H., Dong, C. Y., Kwon, H. S., Sutin, J. D., Kamm, R. D. & So, P. T. (2002) *Biophys J* **82,** 2211-23.
144. Galbraith, C. G., Yamada, K. M. & Sheetz, M. P. (2002) *J Cell Biol* **159,** 695-705.

2 Experimental and analytical methods

Quantitative measurements of cellular biomechanics are based on application of precisely controlled stress (force) or strain (deformation) to single cells and simultaneous measurements of induced cellular deformations with high resolution. In this chapter, I present methods developed for my thesis work to apply forces of up to 10 nN to paramagnetic beads bound to the surface of adherent cells using a single (di-)pole magnetic trap or to alternatively apply constant biaxial strain to cells plated on elastic silicone membranes while monitoring the cells under an optical microscope. In addition, I introduce analytical tools to measure induced magnetic bead displacement and cytoskeletal and nuclear deformations based on digital video recordings taken during the cell manipulation experiments.

2.1 Magnetic trap design and calibration

Magnetic bead microrheology was first introduced to study cellular biomechanics by Bausch et al. (1), who used a single pole magnetic trap to apply tangential forces to 4.5-μm paramagnetic beads bound to the surface of NIH 3T3 fibroblasts. A typical single pole magnetic trap consists of an iron core wrapped with copper wire to create a magnetic coil. The elongated iron core ends in a narrow tip that is brought into close proximity (10-150 μm) to the sample, where it can apply forces in the range from 10 pN to over 10^5 pN to microscopic paramagnetic beads on the cell surface or embedded in the cytoskeleton. The major limitations of single pole magnetic traps are the unidirectional force direction, i.e. force can only be exerted in the direction towards the magnetic trap (pulling), and the rapid decrease of the force field with increasing distance from the magnetic trap, requiring careful calibration of the magnetic trap.

The magnetic trap used in our experiments was designed in collaboration with Hayden Huang and presents a significant improvement over a previous model that was not specifically designed for single pole experiments (2). The main design considerations for our magnetic trap are listed below:

i) Apply high magnetic forces (up to 10 nN) to 4.5-mm paramagnetic beads

ii) Low weight and compact design to enable mounting the magnetic trap on a manual, 5-degree of freedom micromanipulator (MX110R, SD Instruments, OR)

iii) Durable usability without excessively heating of the magnetic trap and the sample

iv) User friendly geometry to provide maximal sample access and operating range for the use with 35 mm polystyrene or glass bottom dishes mounted on a temperature controlled stage on the microscope

v) Magnetic trap tip design to enable consistent force application by bringing the tip into a parfocal position with the sample at a precisely controlled distance

vi) Simple design geometry to reduce manufacturing costs

vii) Long durability of the magnetic trap under experimental conditions, i.e. corrosion resistant material or coating

The overall design geometry was determined by the physical dimensions of the experimental set-up, i.e. the configurations of microscope stage, sample dish, and micromanipulator. In addition, the magnetic field strength and distribution had to be considered in the final design of the magnetic trap. The force acting on a paramagnetic particle subjected to a magnetic field is given by $F = \mu_0 \chi V \nabla (\mathbf{H} \cdot \mathbf{H})$, where μ_0 is the permeability constant, χ the volume susceptibility, V the particle volume, and **H** the external magnetic field strength. Thus, the applied force increases with increasing field strength and increasing field gradient. While the field strength can be controlled by the applied electric current and the number of wire turns, the field gradient is largely determined by the magnetic trap geometry. Computational analysis using the electromagnetic field simulation software Maxwell (Ansoft, Pittsburgh, PA) was performed on a preliminary design to predict the actual magnetic field, and effective force estimates were obtained by comparing the field strength and gradient with values from an earlier magnetic trap model that had previously been force calibrated. The magnetic trap tip was designed to have a flat, square edge that is parallel to the sample

dish surface and perpendicular to the force direction when the magnetic trap is mounted onto the micromanipulator, making distance measurements between the magnetic trap tip and the magnetic beads easier during the experiments. Heat treatment (annealing) and optional nickel plating of the magnetic trap core were included in the manufacturing process to provide good corrosion resistance without affecting biological samples.

The final design of our magnetic trap is presented in figure 2.1 and the detailed design drafts are given in Appendix A. The magnetic trap was manufactured at the MIT central machine shop from a CMI-C rod with high permeability and saturation. The wire wrapping was later added in our laboratory and each layer was secured with epoxy. To avoid excessive heat generation due to resistive heating in the electric wire, the magnetic coil consists of ~300 turns of large diameter wire (0.82 mm^2 cross-section, AWG 18), wrapped in 5 layers of ~60 turns/layer. Consequently, the magnetic trap can be run at the maximum current of 1.5 amp without exceeding a maximal temperature of 31 °C (figure 2.2), still below the regulated temperature in the sample chamber (37 °C). The finished magnetic trap was mounted on a manual micromanipulator and equipped with a Gaussmeter-probe (Lakeshore, Westerville, OH) to record real-time readings of the magnetic field during the experiments. The electric current through the magnetic coil was powered by a computer-controlled amplifier (6545A, Agilent Technologies, Englewood, CO) and regulated using custom-written LabVIEW software (National Instruments, Austin, TX) to allow application of custom-generated forcing functions. The LabVIEW software (Appendix G) was also used to simultaneously record digital videos of the experiments and to store the actual magnetic field strength measured from the Gauss-probe as part of the image data for later analysis.

Magnetic trap calibration was performed for each new magnetic trap and repeated approximately every 6 months to detect possible changes in magnetic trap behavior. Apart from changes associated with tip corrosion and physical abuse of the magnetic trap tip, the inducible force appeared very consistent even over long times. The magnetic trap calibration was performed as previously described (2) and is described in detail below. Paramagnetic beads (4.5-μm Dynabeads M-450, Dynal, Great Neck, NY), identical to the

protein-coated beads used in cell experiments, were suspended in 70% ethanol and air dried onto a 35-mm polystyrene dish. Subsequently, beads were resuspended in dimethylpolysiloxane (DMPS-12M, Sigma) with a kinematic viscosity of 12,500 centistokes. After waiting several minutes to allow residual motions to dissipate, the magnetic trap tip was brought into parfocal position with beads suspended sufficiently far from the fluid surface and the dish bottom to avoid any boundary effects. The magnetic trap was then operated at currents ranging from 0.3 to 1.5 amps while tracking bead positions as they were attracted to the magnetic trap. The force for a given current was computed based on Stokes equation, $F=3\pi\mu D u$, where μ denotes the viscosity of the fluid, D the bead diameter and u the relative velocity between fluid and bead, determined from numerically differentiating the bead position with respect to time. Measurements were repeated for at least three beads at each setting and a regression to the phenomenological power law function $F = a (x + b)^{c}$ was computed as the least square fit, where F denotes the force, x the distance from the magnetic trap and a, b, and c the coefficients to be fitted. Particle tracking and calibration analysis was performed using custom-written MATLAB (Mathworks, Natick, MA) programs listed in Appendix F. Calibration results are presented in detail in chapter 3 (Figs. 3.3 and 3.7). The magnetic force decreases rapidly with increasing distance from the magnetic trap but displays an approximately linear force-current relationship at a fixed distance from the magnetic trap. The maximal force that can be applied at a distance 10 μm from the magnetic trip is ~30 nN, but the magnetic trap is typically operated using magnetic beads at distances of ~100 μm, as the smaller force gradient at this distance reduces the variation in applied force caused by small deviations from the nominal distance.

2.2 Strain device for microscopic observations

As an alternative to controlled force application, one can subject cells to precisely controlled strain and study the cellular response. Cells are plated on elastic surfaces such as fibronectin coated silicone membranes and subjected to uniaxial or biaxial strain of up to 25%. Early experiments commonly investigated cellular morphology and phenotype under uniaxial strain. Experiments with vascular smooth muscle cells (3, 4) or bovine pulmonary artery endothelial cells (5) revealed that cells reorient themselves in response

to applied membrane strain through cytoskeletal remodeling. More recently, application of controlled strain to large cell populations has been used to study mechanotransduction events in cells. Phosphorylation of signaling proteins such as MAPK, induction of mechanosensitive genes, and release of growth factors in response to cyclic biaxial strain have all been demonstrated in various cell types (6-10). Furthermore, microscopic observation of live cells under strain can be used to infer subcellular mechanical properties. The effective strain distribution within intracellular compartments such as the nucleus depends on the relative stiffness of these intracellular compartments in respect to the surrounding cytoskeleton. Measurements on cytoskeletal and nuclear strain distribution in bovine aortic endothelial cells subjected to uniaxial deformations of up to 25% found that cytoskeletal strain closely matches the applied membrane strain, while the nucleus is 5-10 times stiffer than the cytoskeleton, resulting in significantly lower strain within the nucleus (11). Here, we designed a strain device to microscopically monitor subcellular deformations under applied biaxial strain. The main design considerations are listed below:

i) Application of a constant, uniform bi-axial strain field at various strain levels ranging form 0 to 25%

ii) Observation of cells before, during, and after stretch under high magnification using fluorescence and bright field microscopy

iii) Suitable conditions to provide cell growth and attachment prior to and during experiment

The strain device design was based on a previous model used for cyclic biaxial strain application to smooth muscle cells (6) and neonatal cardiac myocytes (7, 10, 12) but was modified to fit on the stage of an inverted microscope (IX-70, Olympus). The stain device consists of three separate parts (Fig. 2.3). The device base is made of an aluminum plate with a circular protrusion at the bottom to securely fit onto the microscope stage. A low friction polytetrafluorethylene-impregnated Delrin platen is located in the center of the base and is used to apply biaxial strain to a silicone membrane. The base has a large central bore to accommodate the microscope objective mounted on an objective extension to compensate for the increased distance of the sample from the microscope

stage. Four vertical steel pins positioned one at each corner of the plate are used to align the dish-holder plate with the base plate. For strain experiments, cells are cultured in custom-made cell culture dishes with a silicone membrane bottom as the culture surface described in detail elsewhere (6). Culture dishes are autoclavable for sterile cell culture conditions and can be maintained in a cell culture incubator to allow sufficient cell growth and attachment. The cell culture dishes are equipped with threads and a wide collar on the outside to securely fit into the dish-holder plate of the strain device in a predetermined position. The dish-holder plate contains four polytetrafluoroethylene bearings, one in each corner, to allow precise alignment with the vertical guidance pins from the base plate and to keep the dish/plate assembly in a horizontal position during strain application. In the resting position, the dish/plate assembly rests with the silicone membrane on the central platen. To apply biaxial strain, weights are used to press down the silicone membrane onto the stationary platen, resulting in a homogeneous and uniform biaxial strain field in the central section of the silicone membrane. Friction between the platen and the membrane is minimized by application of chemically inert, silicone-impermeant grease (Braycote 804, Castrol). Maximal membrane strain is limited by nylon spacers placed on the vertical alignment pins, effectively limiting the vertical displacement of the dish. Actual strain levels for specific spacer combinations are listed in table 2.1, confirming that highly reproducible strain levels of up to ~30% can be achieved with the strain device. Cells can be imaged through the thin silicone membrane using high magnification objectives. Digital images of cell and the silicone membrane are taken before, during, and after strain application and are subsequently analyzed with a custom-written MATLAB program to compute membrane and cellular strains.

2.3 Particle tracking algorithm

A high resolution particle tracking algorithm was used to precisely measure induced magnetic bead displacements and cellular surface deformations. It is important to note that the centroid of an object (e.g. the peak of a Gaussian distribution) can be tracked with resolution far exceeding the optical resolution limit, which is defined as the minimum distance between two objects that still allows resolution into two separate

objects and is given by $d_{\min} = 0.61 \lambda / N \sin(\alpha)$ for light microscopy, where λ is the observation light wavelength, N is the refractive index of the medium surrounding the specimen, and α is the angular aperture, defined as the half-angle of the cone of light entering the objective lens from the specimen. In microscopy, the term $N\ sin(\alpha)$ is replaced by the numerical aperture NA. For a specimen viewed by visible light (λ ~450 nm) using an objective with NA = 0.7, this results in a resolution limit of ~400 nm. In contrast, the resolution for determining the centroid dependents only on the noise level and the intensity of the object (signal), i.e. the number of photons emitted. In bright field applications, the signal-to-noise ratio can be simply optimized by adjusting the light intensity and by using objects that form a contrast with the background, e.g. magnetic beads. In fluorescence imaging, however, signal intensity can become a problem especially when using short exposure times or imaging dim or small objects.

In our experiments, positions of magnetic or polystyrene beads during force application were recorded using high speed (for bright field images) or high sensitivity (for fluorescence images) digital cameras. Digital recordings were subsequently analyzed using custom-written MATLAB software to compute high resolution displacement-time curves for each bead. The particle tracking algorithm consists of a two-dimensional cross-correlation algorithm to track the approximate object position with 1 pixel resolution over time and a subsequent center-of-mass or Gaussian-fit algorithm to compute the centroid position with sub-pixel resolution. For each tracked object, the user can select the approximate object size (needed for the centroid determination), the search area (the area scanned in the cross-correlation algorithm), the image mode (bright field or fluorescence; signals are inverted for bright field images in order to better track dark objects), the centroid computation algorithm (Center-of-mass or Gaussian-fit), and the optional use of threshold filtering to eliminate low level background noise (figure 2.4). Tracking results are recorded in pixel coordinates as comma-separated-value files (*.cmv) and are converted into corresponding metric units by applying spatial conversion factors that have been calibrated for each microscope, camera, and objective combination (see table 2.2). In addition to the centroid position, the program displays peak cross-

correlation values and several additional controls to confirm that the correct object was tracked through the entire time course, as magnetic beads occasionally detach from the cell.

The practical resolution of our tracking algorithm was determined using two independent calibration methods. In the first method, the particle tracking program was applied to synthetically generated image sequences consisting of objects with a Gaussian-signal distribution and varying degrees of random noise. Tracking results were compared with the known correct object positions, yielding the tracking errors listed in table 2.3. A second method was used to estimate the actual particle tracking resolution under experimental conditions. Immobilized magnetic beads air-dried onto a cover slip were imaged using a high speed digital camera (Roper Megaplus ES310 T) and bead positions were analyzed using our particle tracking program. Table 2.4 contains the resolution limits and the maximal positional errors obtained from these experiments. Unlike the first method that yielded insight into the resolution of the computational algorithm itself, results from the second method also depend on the experimental set-up, as the choice of sample, camera, microscope, imaging objective and lighting conditions determine the signal-to-noise ratio recorded on the camera CCD-sensor, and the actual CCD-sensor pixel size will further affect the resolution. Under the experimental conditions routinely used in our experiments, we achieved spatial resolutions of <10 nm. In addition to the high resolution 2-dimensional particle tracking presented here, we also developed a prototype for a 3-dimensional particle tracking system, presented in Appendix C.

2.4 Strain analysis program

Quantitative strain analysis of intracellular domains can provide valuable insight into subcellular mechanical properties, as the relative strain distribution between specific cell domains depends on the contribution of each domain stiffness (see Appendix B). In our experiments, we are particularly interested in nuclear deformation and nuclear mechanical properties. Therefore, it is necessary to compute nuclear and cytoskeletal strain as well as applied membrane strain with high resolution.

Here, we present a strain analysis algorithm to compute local or global strain levels based on the apparent displacement of specific image features matched between pre-strain and strained images, although other time points are also possible (e.g. before and after strain). We found that tracking the displacements of distinct features is more reliable than stain calculations based on changes in feature length or area, as the apparent cross-sectional area/length depends on the vertical focal position when imaging three-dimensional objects. In addition, fluorescence intensity and glare or inappropriate thresholding can further affect apparent feature size or length, while the centroid position of small distinct features is not affected as long as the same features can be identified in both images.

The strain analysis algorithm is based on the idea that distinct image features in the base image (pre-strain) can be mapped into the matching features of the input (strain) image by applying a linear conformal transformation matrix, assuming a constant strain field within the selected image region. The linear conformal spatial transformation represents a combination of scaling, translation and rotation image transformations and is well suited to describe the deformations observed under biaxial strain. Deformations and displacements in the z-direction cannot be resolved using our current methods that focus on images of corresponding two-dimensional cross-sections. Pairs of matching image features are manually selected by the user in each image (i.e. pre-strain and strain) as shown in figure 2.5. To compute global membrane strain, image features are selected that are located directly on the silicone membrane and that are distributed throughout the field of view. Localized nuclear strain on the other hand is calculated by only selecting specific features within the nucleus, such as spots of high chromatin density. The paired image features serve as control point pairs to compute the linear conformal spatial transformation matrix. In the two-dimensional cross-section images, each control point p_i can be represented in its spatial coordinates (x_i, y_i) and written as one row of the control point matrix u as

$$u = \begin{pmatrix} x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \\ \dots & \dots & \dots \end{pmatrix} \tag{1}$$

For a linear conformal spatial transformation, the corresponding positions of control points in the input image (strain), represented by the matrix u', can be computed by applying the transformation matrix T to the control point matrix u of the base (pre-strain) image:

$$u' = uT, \tag{2}$$

where T is the spatial transformation matrix of the form

$$T = \begin{pmatrix} a_{11} & a_{12} & 0 \\ a_{21} & a_{22} & 0 \\ a_{31} & a_{32} & 1 \end{pmatrix}. \tag{3}$$

The spatial transformation matrix can be expressed as a combination of a scaling transformation and a rotation/translation transformation, yielding the following expression

$$u' = \begin{pmatrix} x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \\ \dots & \dots & \dots \end{pmatrix} \begin{pmatrix} s & 0 & 0 \\ 0 & s & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \cos\alpha & \sin\alpha & 0 \\ -\sin\alpha & \cos\alpha & 0 \\ dx & dy & 1 \end{pmatrix}, \tag{4}$$

where s is the scaling constant, α the degree of rotation, and dx and dy represent the translation in x and y respectively. This expression yields the following equation for each control point (x_i, y_i) and its corresponding point (x_i', y_i') in the input image.

$$\begin{pmatrix} x_i' & y_i' & 1 \end{pmatrix} = \begin{pmatrix} x_i & y_i & 1 \end{pmatrix} \begin{pmatrix} s\cos\alpha & s\sin\alpha & 0 \\ -s\sin\alpha & s\cos\alpha & 0 \\ dx & dy & 1 \end{pmatrix}. \tag{5}$$

Comparing equation *5* with equations *2* and *3* reveals the relationship between the general transformation matrix entries a_{ij} and the linear conformal spatial transformation parameters s, a, dx, and dy:

$$\begin{aligned} a_{11} &= a_{22} = s\cos\alpha \\ a_{12} &= -a_{21} = s\sin\alpha \\ a_{31} &= dx \\ a_{32} &= dy \end{aligned} \tag{6}$$

or

$$\tan\alpha = \frac{a_{12}}{a_{11}}$$
$$s = \frac{a_{11}}{\cos a} \tag{7}$$

In the case of biaxial strain application, rotations are generally negligible, i.e. $\alpha \rightarrow 0$ or $a_{12} << a_{11}$ and thus $\cos\alpha \approx 1$, reducing the expression for the scaling factor s to

$$s = a_{11} \tag{8}$$

Assuming a homogenous strain field between the selected control points, the scaling factor s serves as a good estimate for the actual strain in the image plane. A minimum of two pairs of control points is necessary to compute the four independent transformation matrix entries ($a_{11} = a_{22}$, $a_{12} = -a_{21}$, a_{31}, a_{32}). In the case of additional control point pairs, the transformation matrix is computed based on the least square fit solution for all control points. In our experiments, we generally use 8-20 control point pairs with a minimum of 6 pairs to reduce the influence of a single control point on the overall results. In addition, a normalized cross-correlation algorithm with subpixel resolution is used to improve the localization of the manually matched control points. After computing the transformation matrix T, we verify for each experiment that the assumption of small rotations is valid (i.e. $a_{12} << a_{11}$) and we visually compare the original image with the predicted image, i.e. the transformation matrix applied to the base image) to validate the assumption of a linear conformal transformation. The strain analysis algorithm can be applied to any pair of images, i.e. pre-strain/strain, pre-strain/post-strain, etc., and is used to compute membrane strain, cytoskeletal strain and nuclear strain.

2.5 Cell type specific analysis

Adult cardiac myocytes have a highly directional and periodic cytoskeletal organization, consisting of ~2 μm long sarcomeres arranged in series to form myofibrils spanning the length of the cell. Sarcomeres in adjacent myofibrils are aligned through desmin filaments at the Z-disc, resulting in a characteristic striation pattern that can be observed

even at intermediate magnification through the light microscope. The distinct striation pattern found in adult cardiac myocytes can be used to obtain high resolution cytoskeletal displacement maps without the need for fluorescent markers used in other cell types. The high contrast striation pattern provides an ideal target for two-dimensional cross-correlation analysis, delivering displacement maps of the entire cytoskeleton in pixel-resolution (i.e. 300 nm at 30× magnification) or higher. Even better resolution can be achieved by tracking the position of single sarcomeres along the myofibril direction using center-of-mass or Fast-Fourier-Transformation (FFT) algorithms, resulting in a spatial resolution of <20 nm. These methods and their application on adult mouse cardiac myocytes are described in more detail in chapter 5.

setting	strain (%)
1	5.1 ± 0.04
2	10.1 ± 0.18
3	18.2 ± 0.07
4	31.1 ± 0.23

Table 2.1. Strain device calibration. Strain levels achieved with our strain device using four different settings. Applied strain was varied by using different sets of weights and/or nylon spacers to limit the maximal vertical displacement of the strain device. Maximal position refers to the strain device resting on the nylon spacer, while submaximal position describes the static equilibrium between applied weight and elastic membrane strain, i.e. dish not resting on nylon spacer. Setting 1: two nylon spacers, one weight plate (2.5 lb) - one weight plate is sufficient to bring strain device to maximal position; setting 2: one nylon spacer, one weight plate (submaximal position); setting 3: one nylon spacer, two weight plates (maximal position); setting 4: no nylon spacer, two weight plates. This is the absolute maximum strain that can be achieved using the strain device. Results are given as mean ± SEM.

camera	magnification	pixel / μm	nm / pixel
Roper CoolSNAP HQ	10×	1.554	644
	20×	3.086	324
	30×	4.620	216
	40×	6.204	161
	60×	9.569	105
	90×	14.270	70
Roper MegaPlus ES 310T	10×	1.100	909
	20×	2.231	448
	30×	3.331	300
	40×	4.461	224
	60×	6.827	147
	90×	10.227	98

Table 2.2. Spatial calibration. Conversion factors from pixel measurements to corresponding feature size for the cameras/objectives used in our experiments on an Olympus IX-70 microscope. The 30× and 90× magnifications were achieved by using the microscope's 1.5× magnification option in combination with the 20× and 60× objective respectively. Calibration was performed by recording and analyzing images of a calibrated 0.1 mm/50 division graticule (Pyser-SGI, UK) using the ImagePlus software (MediaCybernetics, Silver Spring, MA).

	error	small beads (d=3.5)	medium beads (d=15)	large beads (d=30)
without noise	**mean error**	0.0003	0.0009	0.0009
	max error	0.0013	0.0021	0.0026
random noise	**mean error**	0.0221	0.0218	0.0231
	max error	0.0585	0.0514	0.0747

Table 2.3. Particle tracking resolution for synthetic images. Synthetic images of beads were tracked with the particle tracking program using the center-of-mass algorithm and computed bead positions were compared with the known values. All values are given in pixel units. Conversion factors from pixel to μm can be found in table 2.2. Experiments are typically performed with ~4.5-μm magnetic beads at 30× magnification and recorded using a Roper MegaPlus ES 310T camera, resulting in a corresponding bead diameter of ~15 pixel. Random noise: random noise with a uniform distribution between 0 and 10% of the maximal pixel intensity was added applied to the synthetic image.

objective magnification	tracking resolution		maximal error	
	pixel	nm	pixel	nm
20×	0.016	7.02	0.088	39.26
30×	0.025	7.60	0.129	38.81
40×	0.016	3.67	0.085	18.94
60×	0.012	1.82	0.071	10.42

Table 2.4. Particle tracking resolution under experimental conditions. Beads identical to the ones used in the experiments were immobilized onto a cover slip by air drying and imaged at 60 frames/s on an Olympus IX-70 microscope with a digital camera (Megaplus ES 310T, Roper Scientific). Subsequently, bead centroid positions were computed using the particle tracking program. Tracking resolution was defined as the bead position standard deviation over a 200 frame period. The maximal error was defined as the maximal difference in the apparent position measurement for each object. Values for the x-coordinates are presented as a function of imaging magnification and are expressed as pixel-units and in corresponding physical dimensions (n = 3 for each measurement). Values for the y-coordinate were virtually identical.

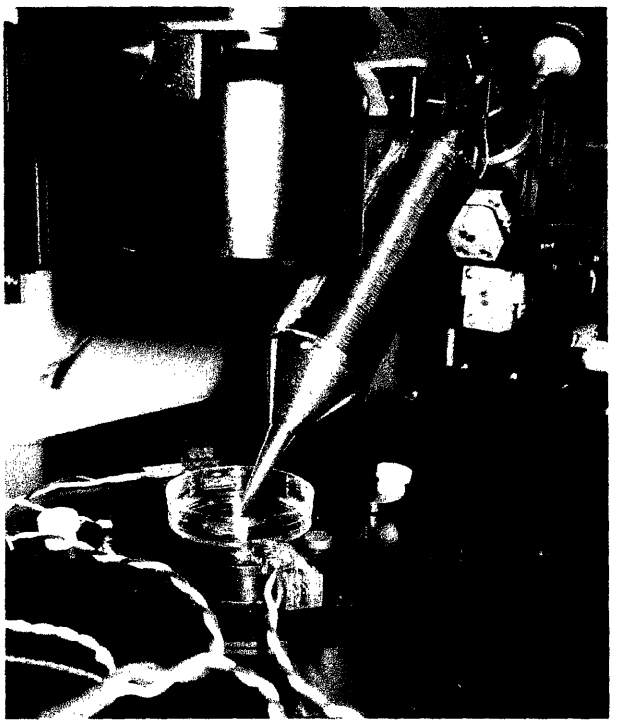


Figure 2.1. Magnetic trap set-up. The magnetic trap is mounted on a manually operated micromanipulator at an angle of ~45°. At this angle, the lower edge of the magnetic trap tip is approximately parallel to the microscope stage. The magnetic coil is powered by a computer controlled power supply not visible in this image. The sample dish on the temperature controlled microscope stage can be seen in the image center.

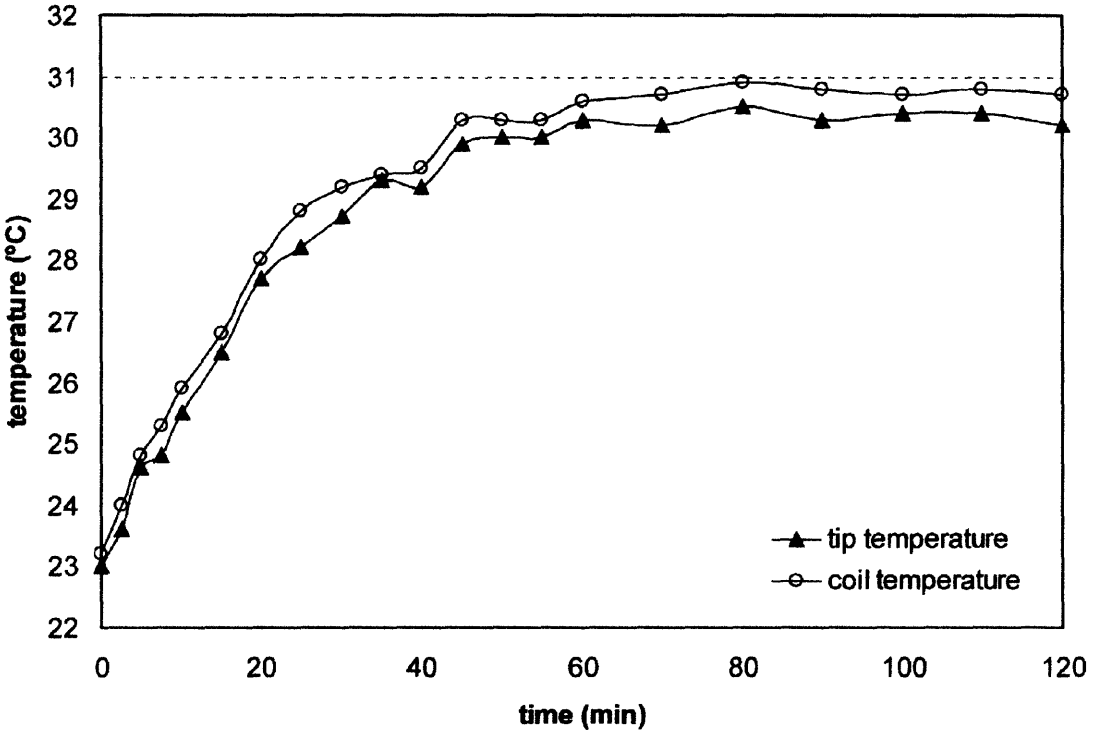


Figure 2.2. Magnetic trap operating temperature curve. Temperature was measured at the magnetic trap tip (▲) and on the surface of the magnetic coil (○) while the magnetic trap was operated at the maximal current of 1.5 amps. Temperatures did not exceed 31 °C (dashed line). Ambient room temperature was 23 °C throughout the entire the time course of the measurements.

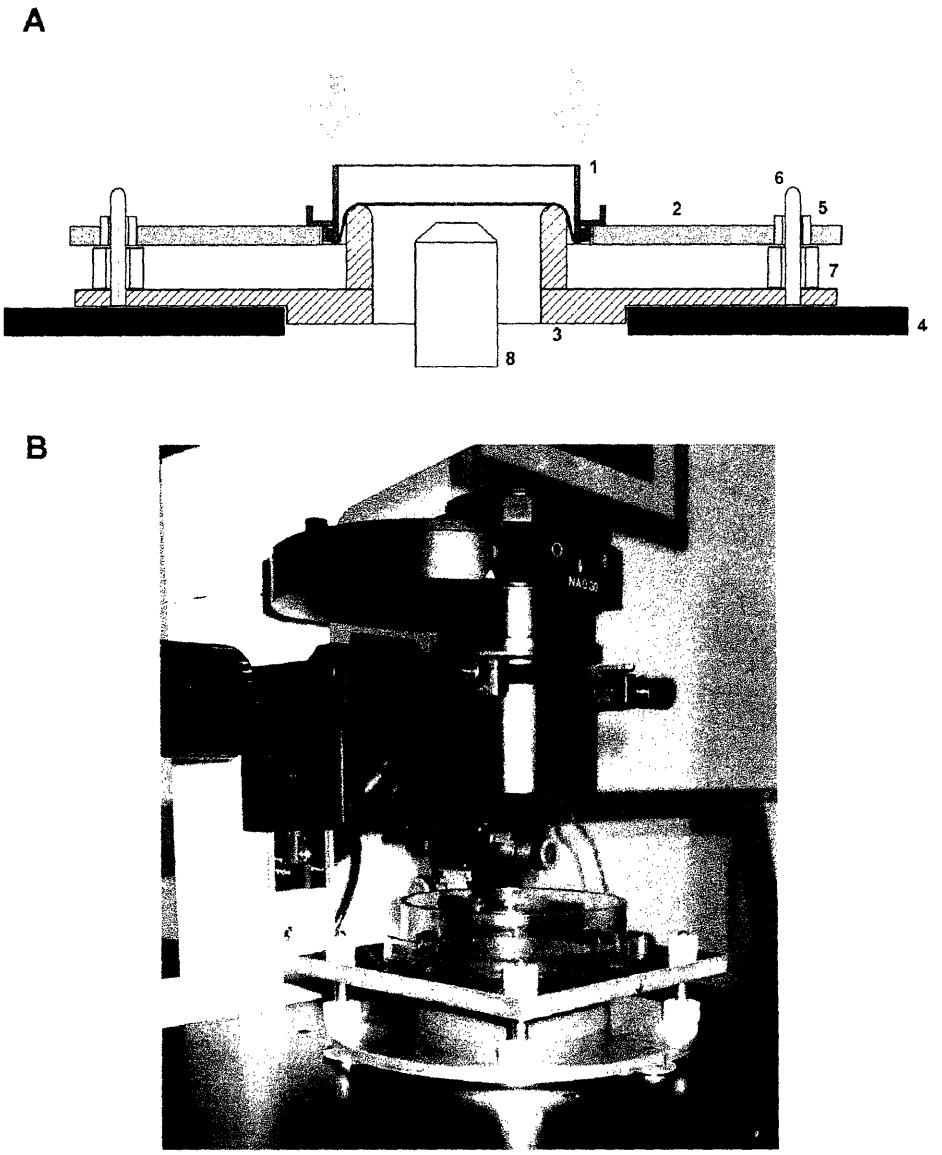


Figure 2.3. Strain device design. (A) Schematic of the strain device in cross-section. The silicone membrane cell culture dish (*1*) is mounted onto the dish-holder plate (*2*) and placed on the base plate (*3*) that fits firmly onto the microscope stage (*4*). Polytetrafluoroethylene bearings (*5*) in the dish-holder plate and vertical pins (*6*) on the base plate provide precise alignment and stabilization of the silicone membrane dish relative to the stationary platen. Nylon spacers (*7*) limit the vertical displacement and thus the applied membrane strain. A central bore in the base plate accommodates the high power microscope objective (*8*). (B) Strain device with a cell culture dish in resting condition mounted on an inverted microscope.

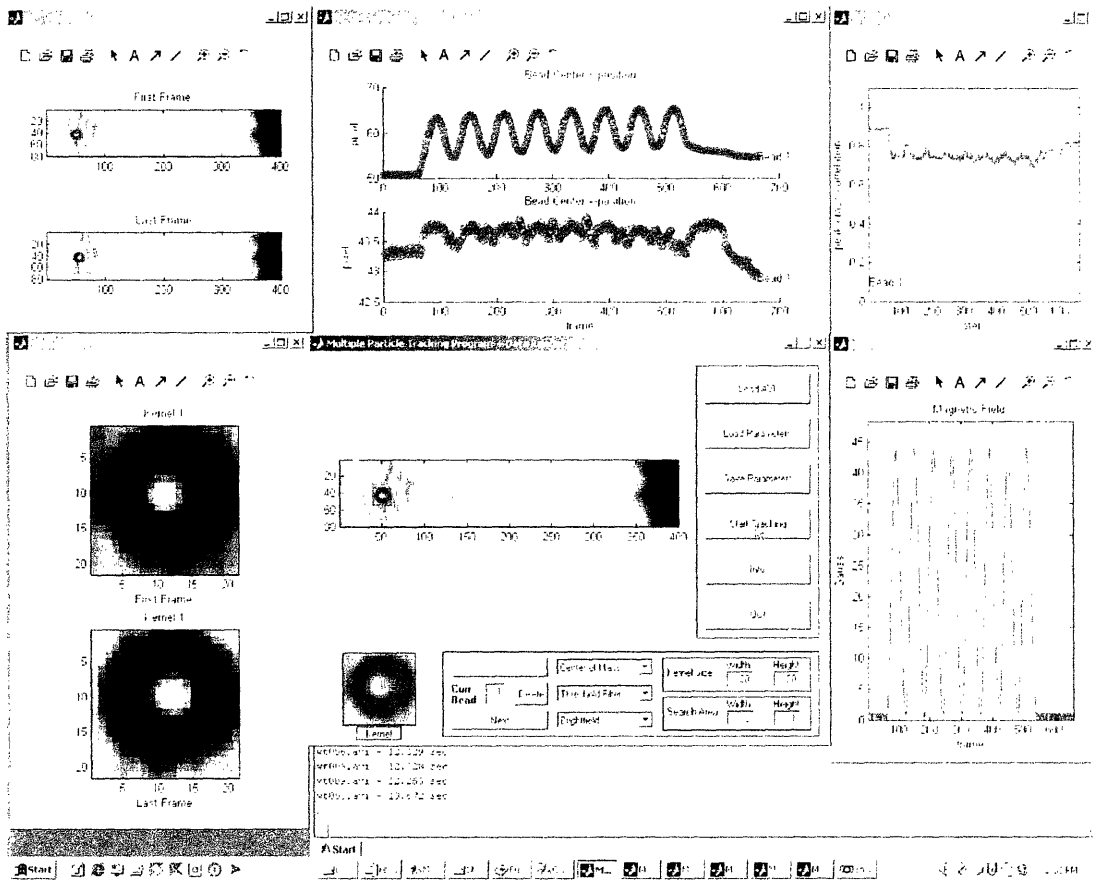


Figure 2.4. Screenshot of the particle tracking program. The window in the image center contains the graphical user interface for the multiple particle tracking program. Displayed is the first frame of the digital video of the bead experiments with one bead selected for tracking (blue square; up to 99 beads can be tracked simultaneously). A close-up of the selected bead and the computed centroid position are displayed in the bottom left corner of the window. Other features include menus to select the desired parameters for the particle tracking algorithm as explained in the main text. Additional windows display the magnetic bead displacement in x-direction and y-direction (*top center*), the cross-correlation maxima for each frame (*top right*), the magnetic field data (*bottom right*). The two windows on the left provide additional controls over the tracking results by displaying the tracked bead position in the first and last frame (*top left*) and showing close-ups of the magnetic bead in the first and last frame based on the cross-correlation results (*bottom left*). The MATLAM prompt (*very bottom*) lists the last processed video files.

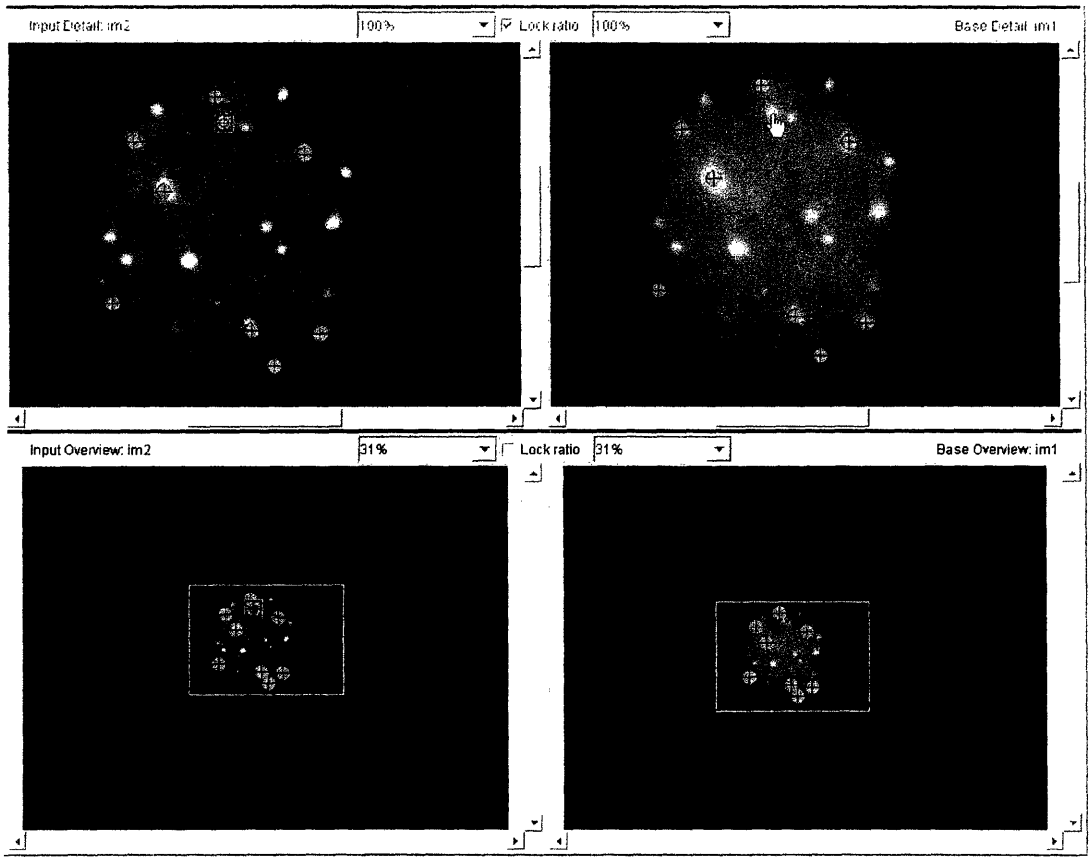


Figure 2.5. Screenshot of the strain analysis program. Shown is a pair of images of a fibroblast nucleus with fluorescently labeled chromatin before (*right*) and during (*left*) strain. Distinct spots of high chromatin density are clearly visible and are used as control points for the strain calculations. The user manually selects control point pairs (green circles) of matching features in each image which are subsequently used to compute the best-fit linear conformal transformation matrix. Generally 8-20 control point pairs are selected for each nucleus. The upper panels provide a zoom function to position the control points more accurately.

2.6 References

1. Bausch, A. R., Ziemann, F., Boulbitch, A. A., Jacobson, K. & Sackmann, E. (1998) *Biophys J* **75,** 2038-49.
2. Lammerding, J., Kazarov, A. R., Huang, H., Lee, R. T. & Hemler, M. E. (2003) *Proc Natl Acad Sci U S A* **100,** 7616-21.
3. Dartsch, P. C. & Hammerle, H. (1986) *Eur J Cell Biol* **41,** 339-46.
4. Barbee, K. A., Macarak, E. J. & Thibault, L. E. (1994) *Ann Biomed Eng* **22,** 14-22.
5. Gorfien, S. F., Winston, F. K., Thibault, L. E. & Macarak, E. J. (1989) *J Cell Physiol* **139,** 492-500.
6. Cheng, G. C., Briggs, W. H., Gerson, D. S., Libby, P., Grodzinsky, A. J., Gray, M. L. & Lee, R. T. (1997) *Circ Res* **80,** 28-36.
7. Wang, Y., De Keulenaer, G. W., Weinberg, E. O., Muangman, S., Gualberto, A., Landschulz, K. T., Turi, T. G., Thompson, J. F. & Lee, R. T. (2002) *Am J Physiol Heart Circ Physiol* **283,** H533-9.
8. Davis, R. J. (1993) *J Biol Chem* **268,** 14553-6.
9. Sadoshima, J. & Izumo, S. (1993) *Embo J* **12,** 1681-92.
10. Feng, Y., Yang, J. H., Huang, H., Kennedy, S. P., Turi, T. G., Thompson, J. F., Libby, P. & Lee, R. T. (1999) *Circ Res* **85,** 1118-23.
11. Caille, N., Tardy, Y. & Meister, J. J. (1998) *Ann Biomed Eng* **26,** 409-16.
12. De Keulenaer, G. W., Wang, Y., Feng, Y., Muangman, S., Yamamoto, K., Thompson, J. F., Turi, T. G., Landschutz, K. & Lee, R. T. (2002) *Circ Res* **90,** 690-6.

3 Tetraspanin CD151 regulates α6β1 integrin-dependent mechanical force transduction

Externally applied forces or strains are transmitted to the cytoskeleton through adhesion receptors on the cell surface (e.g. integrins) that simultaneously bind to cytoskeletal proteins and to extracellular matrix ligands. At the same time, contractile forces generated in the cytoskeleton are transmitted through these adhesion receptors to the extracellular matrix. Mutations in proteins involved in the adhesion complex could lead to differences in adhesion strengthening or mechanical coupling to the cytoskeleton, therefore affecting cellular responses to mechanical stimulation. To study the role of this mechanical coupling between the cytoskeleton and the extracellular matrix on mechanotransduction, we applied the methods described in the previous chapter to measure adhesion strength and cytoskeletal coupling in mutant and wild-type cells. The results of these experiments have been published in *Proc Nat Acad Sci* 2003, vol. 100(13), pages 7616-7621, and are described below.

3.1 Introduction

Adhesion receptors in the integrin family bind simultaneously to extracellular matrix ligands and to cytoskeletal proteins, thereby transducing external biomechanical stimuli into internal biochemical responses. Biomechanical forces mediated through integrins regulate cell migration, extracellular matrix assembly and remodeling, wound healing, and tissue morphogenesis (1-5). The application of defined forces upon direct engagement of specific integrins with fibronectin (6), laminin (7) or antibody to the integrin β1 subunit (8) results in strengthening of local cytoskeletal linkages. Also, agents such as thrombin may indirectly induce stimulation of integrin-cytoskeletal stiffness, as measured using fibronectin-coated magnetic beads (9). Because different extracellular matrix protein ligands trigger distinct integrins, coupled to distinct signaling pathways (10-12), mechanisms for regulating cell adhesion-related events could vary considerably. For example, adenocarcinoma cells adhering to fibronectin preferentially develop stress fibers and focal contacts, whereas the same cells adhering to laminin form lamellipodia (10-12). Such results suggest that laminin-binding and fibronectin-binding

integrins could have fundamental mechanistic differences. In this regard, only the laminin-binding integrins ($\alpha3\beta1$, $\alpha6\beta1$, $\alpha6\beta4$, $\alpha7\beta1$) show strong lateral association with CD151, a transmembrane protein in the tetraspanin family (13-16). Monoclonal antibody perturbation studies indicate that CD151 modulates integrin-dependent migration, neurite outgrowth, and cell morphology on matrigel (17-20). The short C-terminal cytoplasmic domain of CD151 was particularly important for $\alpha6\beta1$ integrin-mediated spreading, migration, and cellular cable formation on matrigel (20). Besides CD151, several other members of the tetraspanin protein family (such as CD9, CD81, and CD63) also regulate integrin-dependent cell migration. Although tetraspanin proteins may associate with signaling enzymes and regulate signaling pathways (14, 21-23), the mechanisms whereby they affect cell migration and spreading have not been established.

The preponderance of evidence suggests that CD151 and other tetraspanins do not modulate integrin-dependent static cell adhesion (22). Because CD151 strongly influences $\alpha6\beta1$ integrin-dependent cellular cable formation on Matrigel (20), we hypothesized that CD151 could be regulating strengthening of adhesion mediated through the integrin. To test this hypothesis directly, we engaged $\alpha6\beta1$ integrin with laminin-coated beads, exerted a defined mechanical force on the beads, and analyzed bead responses in terms of bead detachment. Magnetic traps have been used previously to explore mechanical properties of cells and to study mechanotransduction events (8, 9, 24-30). Here, we extend the range of magnetic trap application to include probing the adhesion strength of beads coated with either laminin or fibronectin to integrin receptors on the cell surface. Results from this application demonstrate clearly that CD151 plays a key role in regulating the time-dependent gain of adhesion strength for the $\alpha6\beta1$ integrin, with the C-terminal region of CD151 being particularly important.

3.2 Materials and methods

Cells and antibodies. The CD151 C(217)-N mutant was generated (31), and later renamed as CD151-c2-NAG2 (20). For this mutant, the C-terminal HLRVIGAV-GIGIACVQVFGMIFTCCLYRSLKLEHY of CD151 was replaced by the corresponding NLLAVGIFGLCTALVQILGLTFAMTMYCQVVKADTYCA from NAG2 tetraspanin

protein. For stable expression of wild-type CD151 (CD151-WT) and mutant CD151, plasmid DNAs were transfected into NIH 3T3 cells using Lipofectamine (Life Technologies, Grand Island, NY). After 48hrs, cells were cultured in media containing 200 μg/ml Zeocin (Invitrogen). After 2 weeks of selection, colonies were pooled and CD151-positive cells were sorted by flow cytometry. Surface expression of CD151 and α6 integrin on NIH 3T3 transfectants was assessed by flow cytometry as described (32). Cells were maintained in Dulbecco's modified Eagle's medium (DMEM, GIBCO) with 10% fetal calf serum and antibiotics.

Cell cable formation and adhesion assays. To observe cellular cable formation, cells were plated on a thick layer of Matrigel in 5% fetal bovine serum/DMEM at 5×10^4 cells per well in a 24-well plate, analyzed by using a ZEISS Axiovert 135 microscope, and photographed after 18 h as described (20). To measure static cell adhesion, cells were incubated on laminin-1 (coated at 20 μg/ml) for varying times, washed, and attached cells were quantitated using the Cytofluor 2300 measurement system (Millipore, Bedford, MA) as described (33).

Particle tracking and magnetic trap calibration. Cells and magnetic beads were imaged through an inverted light microscope (IX-70; Olympus) at ×20 and ×30 magnification. Images were recorded with a Megaplus ES310/T digital camera (Roper Scientific MASD, San Diego) at 25 frames per sec and stored on a computer. Custom-written MATLAB software was used to track bead position with a spatial resolution of ≈30 nm at ×30 magnification. The magnetic trap was calibrated by suspending 4.5-μm diameter magnetic beads (Dynabeads M-450; Dynal, Great Neck, NY) in dimethylpolysiloxane (DMPS–12M; Sigma) and tracking bead positions as they were attracted to the magnetic trap operated at a range of electrical currents (0.3 to 1.5 amps). The force for a given current was computed based on Stokes Law, $F=3\pi \nu D u$, where ν denotes the viscosity of the fluid, D the bead diameter and u the relative velocity between fluid and bead, determined from numerically differentiating bead position with respect to time. Measurements were repeated for at least three beads at each setting and an exponential regression of type $F = a (x + b)^c$ was computed as the least square fit, where

F denotes force, x distance from the magnetic trap and a, b, and c the coefficients to be fitted.

Bead coating and measurements of detachment and displacement. Magnetic beads (M-450, Dynal) were coated for 18 hours at 37°C with laminin (L-2020, Sigma) in acetate buffer (pH 4.0) or fibronectin (33016-023, GIBCO/BRL) in borate buffer (pH 8.5), with each protein at 500 μg/ml. Beads were also coated with anti-α6 monoclonal antibody (555734, Pharmingen). Cells were plated in DMEM with 10% FCS, penicillin/streptomycin and Zeocin (200 μg/ml) at 3 ml per dish on gelatin-coated (0.1% gelatin in PBS, overnight at 4°C) 35-mm polystyrene cell culture dishes (430588, Corning) at a density of 150,000 cells per dish and incubated at 37°C overnight. The medium was replaced the next day with DMEM containing 5% FCS and 3-5 μl of laminin-coated, fibronectin-coated, or anti-α6 mAb-coated magnetic bead suspension (5×10^5 beads/dish) and incubated at 37°C for 120 min to guarantee sufficient bead attachment. On a temperature-controlled stage, cells were imaged at ×30 magnification by using an inverted light microscope (IX 70, Olympus). Nonconfluent cells with a single bead firmly attached (confirmed by a lack of bead motion) were selected for magnetic trap experiments. The magnetic trap was brought into a parfocal position 115 μm from the bead. Digital video acquisition was started while electric current powering the magnetic trap was increased from 0 to 1.5 amps in steps of 0.3 amps every two sec (95% of detachment events occur within 1.8 sec). Video acquisition was continued for 2 sec after termination of force application (see Fig. 3D) to monitor bead relaxation. After force application, a new cell was selected at least 5 mm away from any previous force application sites to avoid studying preconditioned cells. Fifteen cells were selected in each dish, and experiments were concluded within 30 min per dish. Bead displacement and detachment were evaluated offline using digitally recorded videos. Maximal displacement was defined as the difference in mean bead position between the last 10 frames (= 0.4 sec) of force application at each force level and the initial position, estimated as the mean bead position during the last 25 frames (= 1 sec) before the start of force application.

Antibody measurements. CD151-WT cells were plated as described in the protocol for detachment studies. The next day, media was replaced with DMEM containing 5% FCS and 8 μl of laminin-coated magnetic bead suspension (final concentration of 8×10^5 beads per dish) in the absence or presence of purified anti-human integrin α6 antibody (Pharmingen, 555734, mAb GoH3, final concentration of 5-10 μg/ml) and incubated at 37° C for ≈120 min. Detachment measurements were carried out as described above.

Attachment measurements. Cells were plated according to the protocol given for detachment experiments, but at a higher density (≈300,000 cells/dish) to achieve a confluent monolayer of cells. On the next day, media was replaced with DMEM containing 5% FCS and magnetic beads at a final concentration of 5×10^5 beads per dish. The cells were then incubated at 37° C for 120 min. Bead attachment was measured by subjecting a large section of cells to low forces (<0.6 nN) for 4 sec and counting the fraction of beads that remained firmly attached during that time. In contrast to the detachment study, we did not select for firmly attached beads but included all beads located on the cell surface. For each dish, 10-12 sections were evaluated, with a distance of at least 5 mm between sections. Experiments were repeated on at least two dishes per cell type for a total number of ≈100-200 cells.

Statistics. Statistical analysis was performed by using the INSTAT software (GraphPad, San Diego). Differences in adhesion and detachment events were evaluated using a 2-by-2 contingency table and applying Fisher's exact test. For the displacement measurements, the non-parametric Mann-Whitney test was used to detect differences in median bead displacement, while an unpaired *t* test (allowing different standard deviations) was used for the log-transformed data. Results are expressed as mean ± standard error. A two-tail *P* value of <0.05 was considered significant.

3.3 Results

Characterization of CD151 mutant. Previously we demonstrated that deletion or exchange of the CD151 C-terminal cytoplasmic tail region did not alter cell surface expression, CD151-integrin association, or integrin-dependent static cell adhesion to

laminin-1. However, such mutations did markedly impair α6β1 integrin-dependent cell spreading and formation of cellular cables (20). Confirming and extending those results, we show here that mutant CD151-c2-NAG2 (human CD151 with C-terminal region replaced by corresponding sequence from NAG2 tetraspanin) failed to support cellular cable formation by NIH 3T3 cells when plated on Matrigel (Fig. 1 *Bottom*). In contrast, NIH-3T3 cells that were mock transfected (Fig. 1, *Top*), or expressing CD151-WT (*Middle*), showed abundant cable formation. Nearly complete inhibition by anti-α6 mAb GoH3 (Fig. 1, *Left*) and strong inhibition by anti-human CD151 mAb (Fig. 1, *Center*) confirm that cellular cable formation is dependent on the α6β1 integrin, and on human CD151, when it is present. In the absence of human CD151, NIH-3T3 cells were not affected by anti-CD151 mAb 5C11 (*Top* and *Middle*). As seen previously (20), the CD151-c2-NAG2 mutant (present at 2-3 fold above endogenous CD151) is likely exerting a dominant negative effect on the function of endogenous murine CD151.

CD151-WT and mutant CD151 supported similar levels of static cell adhesion to laminin-1 (at 15 min or 40 min; Fig. 2*A*). This adhesion was almost entirely dependent on the α6β1 integrin, as evidenced by the strong inhibitory effects of mAb GoH3 on all NIH-3T3 cells, regardless of presence of human wild type or mutant CD151 (Fig. 2*A*). As measured by flow cytometry, surface levels of mutant and wild type human CD151 were comparable (Fig. 2**B**, *Right*) as were levels of endogenous α6 in the same cells (Fig. 2*B*, *Left*). As seen previously, the CD151-c2-NAG2 mutant retained full association with laminin-binding integrins α6β1 (in NIH3T3 cells) and α3β1 (in other cells) (20, 34).

Magnetic trap calibration. To investigate possible influence of CD151 on integrin-dependent mechanical force transduction, we used a magnetic trap (Fig.3*A*). The magnetic force applied to a single paramagnetic bead is proportional to the product of the magnetic field and its gradient. Therefore, the force depends on the electric current powering the magnetic trap and on the distance of the bead from the pole tip, as the magnetic field decays exponentially with the distance from the tip (Fig. 7*A*). At a given distance from the tip of our magnetic trap (Fig. 3*A*), the force varies almost linearly with the applied current (Figs. 3*B* and 7*B*). For example, when the magnetic trap was

positioned 115 μm away from the bead and operated at electric currents of 0.3, 0.6, 0.9, 1.2, and 1.5 amps, this resulted in precisely controlled linear forces of 0.24 ± 0.002 nN, 0.57 ± 0.03 nN, 0.85 ± 0.003 nN, 1.23 ± 0.04 nN, and 1.42 ± 0.04 nN, respectively, on a single 4.5-μm paramagnetic bead. Nonconfluent cells with single laminin-coated or fibronectin-coated magnetic beads attached to the cell surface were selected for magnetic trap experiments and firm attachment of bead to cell was confirmed (Fig. 3*C*). The bead was then subjected to a magnetic force that increased from 0 to nearly 1.5 nN in steps of 0.3 nN, using a step duration of 2 sec. Bead position and attachment were continuously monitored during the force application (Fig 3*D*). To avoid possible preconditioning effects of bead-cell attachment, subsequent beads were selected at least 5 mm away from previous force application sites. These results establish that bead detachment and bead displacement can be measured in response to carefully defined forces.

CD151 mutation effects on bead detachment. To assess CD151 C-terminal mutation effects on mechanical force transmission through the α6β1 integrin, we added beads coated with laminin to CD151-2c-transfected NIH-3T3 cells, waited 2 h, and then measured bead detachment. In an initial experiment (at a force = 1 nN), the majority of laminin-beads (24 of 37) detached from CD151-2c-NAG2 transfectants, whereas only a few (5 of 43) detached from mock-transfected NIH-3T3 cells (Fig. 4*A*). In a control experiment, only a few beads coated with fibronectin (which engages the α5β1 integrin) detached from either CD151-2c-NAG2 (5 of 28) or mock (2 of 21) transfectants (Fig. 4*B*), indicating that the CD151-2c-NAG2 mutation was selectively affecting laminin bead detachment. A more comprehensive experiment was then carried out (Fig. 4*C*), over a range of forces (0-1.5 nN). Again the CD151-2c-NAG2 mutant showed significantly easier detachment of laminin-coated beads. This was especially evident for forces of 1.2 nN and higher ($P < 0.001$, Fig. 4*C*). Mock-transfected and CD151-WT transfected NIH 3T3 cells showed comparable bead detachment across the whole range of forces. Among the three transfectants, no significant differences in bead detachment were observed for fibronectin-coated beads at any force level (Fig. 4*D*).

To ascertain the extent to which resistance to laminin-bead detachment is α6β1 integrin-dependent, CD151-WT transfectants were incubated with laminin-beads in the presence of purified anti-murine α6 antibody (5-10 μg/ml) and then bead detachment was measured. As indicated (Fig. 5*A*), bead detachment in the presence of anti-α6 antibody was significantly enhanced over the entire 0-1.5 nN force range. The difference between the two curves defines the magnitude of bead detachment that depends on the α6β1 integrin. Remarkably, the magnitude of this antibody effect (Fig. 5*A*) is very similar to the magnitude of the CD151-tail mutation effect (Fig. 4*C*).

To confirm that the difference in bead detachment is caused by a lack of adhesion strengthening and not a deficiency in initial bead attachment, we measured the adhesion of laminin-coated and fibronectin-coated beads to the transfected cells. Bead adhesion was defined as the fraction of beads located on the cell surface that withstood a weak force application (0.3 – 0.6 nN) for 4 sec. This force level was sufficient to wash off undetached beads, but insufficient to detach beads that had firmly attached to the cell surface (compare with results of Fig. 4*C* and *D*). The C-terminal mutation did not affect adhesion of laminin-coated beads as attachment fractions were comparable for CD151-c2-NAG2 and CD151-WT with the mock-transfectants exhibiting a slightly higher attachment fraction ($P < 0.05$) compared to both c2-NAG2 and CD151-WT (Fig. 5*B*). Attachment of fibronectin-coated beads also was uniformly high with no significant differences among CD151 transfectants (Fig. 5*B*). Beads coated with BSA showed no attachment to any of the cells (data not shown).

Whereas bead detachment was significantly different for CD151-WT and CD151 mutant cells after 2 h (Fig. 4*C*), at an earlier time point (beads attached for 30 min instead of 2 h), CD151-WT and CD151 mutant cells showed a similar ease of detachment of laminin-coated beads (Fig. 6*A*, with key curves from Fig. 4*C* superimposed). In contrast to the laminin-coated beads, anti-α6 antibody-coated beads did not detach very readily at 2 h (Fig. 6*B*) or 30 min (data not shown), and this low level of detachment was not affected by mutant CD151 (Fig. 6*B*).

Increased bead displacement for CD151 mutant. Beads remaining attached could nonetheless show variable bead displacement. To assess whether connection between the extracellular matrix and the cytoskeleton is impaired in CD151 C-terminal mutants, we measured the maximal displacement of magnetic beads during force application. To avoid artifacts such as bead tethering or partial detachment associated with high force levels, we limited our evaluation to bead displacements at the low force level of 0.3 nN. The C-tail exchange mutation (CD151-c2-NAG2) resulted in a significantly increased median bead displacement for laminin-coated beads (0.32 μm vs. 0.20 μm, $P = 0.0022$, see Fig. 8*A* and *C*). No significant difference in bead displacement was observed for fibronectin-coated beads (0.15 μm vs. 0.10 μm, $P = 0.131$, see Fig. 8*B* and *D*).

3.4 Discussion

Our results establish that CD151 plays a key role as a regulator of α6β1 integrin adhesion strengthening. Mutation of the CD151 C-terminal region had little or no effect on static cell adhesion to laminin-1, bead attachment, or laminin-coated bead detachment after bead attachment for only 30 min. However, after beads had attached for 2 h, the CD151 mutant did not show nearly as much adhesion strengthening as CD151-WT. This was manifested as increased detachment differential between mutant and wild type CD151 (especially at 0.9-1.5 nN), with mutant beads being more easily detached. Bead displacement was also significantly increased for mutant CD151, consistent with decreased cellular stiffness. Since the cytoskeleton is the main contributor to cellular stiffness, we conclude that our mutation of CD151 has disrupted dynamic cytoskeleton-dependent processes, critical for α6β1 adhesion strengthening. Previous results (17) showed that CD151 associated strongly with α6β1 (a laminin receptor), but not α5β1 (a fibronectin receptor). Those results are entirely consistent with our CD151 mutation affecting detachment and displacement of laminin-1 beads, but not fibronectin beads. We predict that CD151 should also regulate adhesion strengthening through the α3β1, α6β4 and α7β1 integrins, since those integrins also associate strongly with CD151 (16). It remains unclear why laminin binding integrins, but not fibronectin binding integrins or other integrins would need a tetraspanin protein to augment their adhesion strengthening capabilities.

Elsewhere we observed CD151-dependent cell spreading and cellular cable formation on Matrigel, but the role of CD151 was not clear. Our data now support a "transmembrane linker" model in which (*i*) the integrin contacts laminin, (*ii*) CD151 forms an extracellular contact with the α subunit of laminin-binding integrins (31, 35), and (*iii*) CD151 uses its short cytoplasmic tail to engage as yet unidentified membrane-proximal elements. These unknown elements are critical for strengthening integrin-mediated adhesion, such that it withstands larger mechanical forces. By mutating the CD151 C-terminal region, adhesion strengthening is impaired (this study), and cells cannot spread or form cables (20), likely because adhesion strengthening is needed to transmit the mechanical forces that play an essential role in the process of integrin-dependent cellular cable formation (36, 37) and cell spreading. Consistent with our transmembrane linker model, mutation of either the CD151 C-terminal region, or an extracellular "QRD" site needed for integrin association, resulted in severely impaired cellular cable formation (20, 34). Thus the extracellular CD151 integrin-association site may be just as important as the intracellular C-terminal site with respect to adhesion strengthening, although this remains to be tested in the magnetic bead assay. Providing further support for our transmembrane linker model, inhibition of extracellular integrin-ligand binding by an anti-α6 mAb, and mutation of the intracellular CD151 C-terminal region yielded remarkably similar bead detachment profiles over a range of increasing forces. If we accept that the antibody inhibition experiment (Fig. 5*A*) roughly defines the window of bead detachment that is integrin-dependent, then by comparison (Fig. 4*A*), the C-terminal region of CD151 is at least as critical as, and perhaps even more critical than, antibody-sensitive extracellular contacts. Stated another way, the entire window of bead detachment that is α6 integrin-dependent appears to be eliminated upon C-terminal tail deletion of CD151.

Historically, studies of integrin transmembrane linker functions have focused on cytoplasmic tails of integrins themselves. Our results now expand the focus to include the C-terminal tail of CD151. Elsewhere, three different types of CD151 C-terminal mutations were shown to eliminate cellular cable formation (20). As confirmed here,

exchange of the C-terminal region with a corresponding region from tetraspanin NAG2 eliminated cable formation and adhesion strengthening. A shorter exchange with the C-tail of tetraspanin A15/TM4SF6, or a deletion of the CD151 tail also abolished cable formation (20). Here we have carried out magnetic bead studies for only one CD151 mutant, but we expect that other CD151 tail mutations that disrupt cellular cable formation and cell spreading should also affect detachment and displacement of laminin-coated beads. Previous and current results focus attention on the C-terminal 7 amino acids of CD151 (SLKLEHY) as being critical for adhesion strengthening. It remains to be seen what other proteins might be specifically engaged by this region of CD151. The CD151 molecule could potentially modulate strength of adhesion by affecting integrin clustering. Because CD151 mutations have not altered detachment of anti-α6 antibody-coated beads, static cell adhesion, or early (30 min) bead detachment, we do not suspect alterations in constitutive integrin clustering. However, CD151 tail mutation could affect adhesion strengthening by altering the dynamic integrin clustering that occurs after contact with laminin-coated beads. In this regard, it is ligand occupancy plus clustering rather than clustering alone that promotes recruitment of cytoskeletal proteins to the α5β1 integrin (38).

Results were obtained by using an assay that measures magnetic bead detachment and displacement. Magnetic bead experiments have been previously used to measure the mechanical properties of the cytoskeleton and the cell surface (8, 9, 24-26, 28, 30) and to apply mechanical stimulation to study mechanotransduction events and signaling (27, 29). Others observed (26) that vinculin-deficient cells showed increased bead displacements compared to wild-type cells, demonstrating the capabilities of the magnetic trap system to measure the changes in stiffness of the transmembrane integrin linkages to the cytoskeleton associated with specific proteins. Here, we use high resolution (<10 nm) single-particle tracking at very low force levels to detect small differences in the mechanical coupling between extracellular matrix proteins and the cytoskeleton caused by a mutation in the CD151 tetraspanin. In addition, we further extend the range of magnetic trap applications. We measure the adhesion strength and cellular mechanics of the cell/extracellular matrix interaction by applying a wide range of

forces (0 – 1.5 nN) and measuring bead detachment or displacement as a function of applied force. In the study of cell adhesion, this approach differs from other quantitative cell attachment methods that are designed to integrate adhesion results from many cells at once (39, 40).

Techniques such as atomic force microscopy, micropipette aspiration or optical traps have been traditionally used for the study of single integrin-ligand molecular adhesion forces in the range of 30-150 pN (41-45). By contrast, our method can apply much higher force levels (0-1.5 nN), well suited to study the concerted action of many adhesion receptors simultaneously. At present, there is no evidence that tetraspanins can modulate adhesion forces for individual integrin heterodimers. Rather, the current results are consistent with CD151 affecting adhesion strengthening that is mediated through many integrins (e.g. through effects on cytoskeletal linkages and/or integrin clustering). Notably, the magnitude of cellular force regulated by CD151 (1-1.5 nN) is consistent with the ≈1-5 nN forces exerted by keratocytes during cell migration (46) and fibroblasts during collagen gel contraction (47). Hence, we can now better understand why CD151 mutation caused alterations in cell migration, spreading, and cellular cable formation.

In conclusion, we have demonstrated that mutation of the C-terminal region of tetraspanin CD151 markedly reduces integrin α6β1 adhesion strengthening. The concept of tetraspanins as regulators of adhesion strengthening emerges as a useful new paradigm to help explain many previous examples in which tetraspanins were shown to regulate cell migration, spreading, morphology, and cellular cable formation on Matrigel basement membrane. Our results focus attention on the CD151 cytoplasmic tail as being uniquely important for modulation of cytoskeletal engagements. Notably, CD151 complexes with laminin-binding integrins are present in a variety of tissue locations including smooth muscle costameres (16) and endothelial cell-cell junctions (18, 48). Thus, CD151 modulation of adhesion strengthening could play a key role in force-dependent processes such as muscle contraction and angiogenesis.

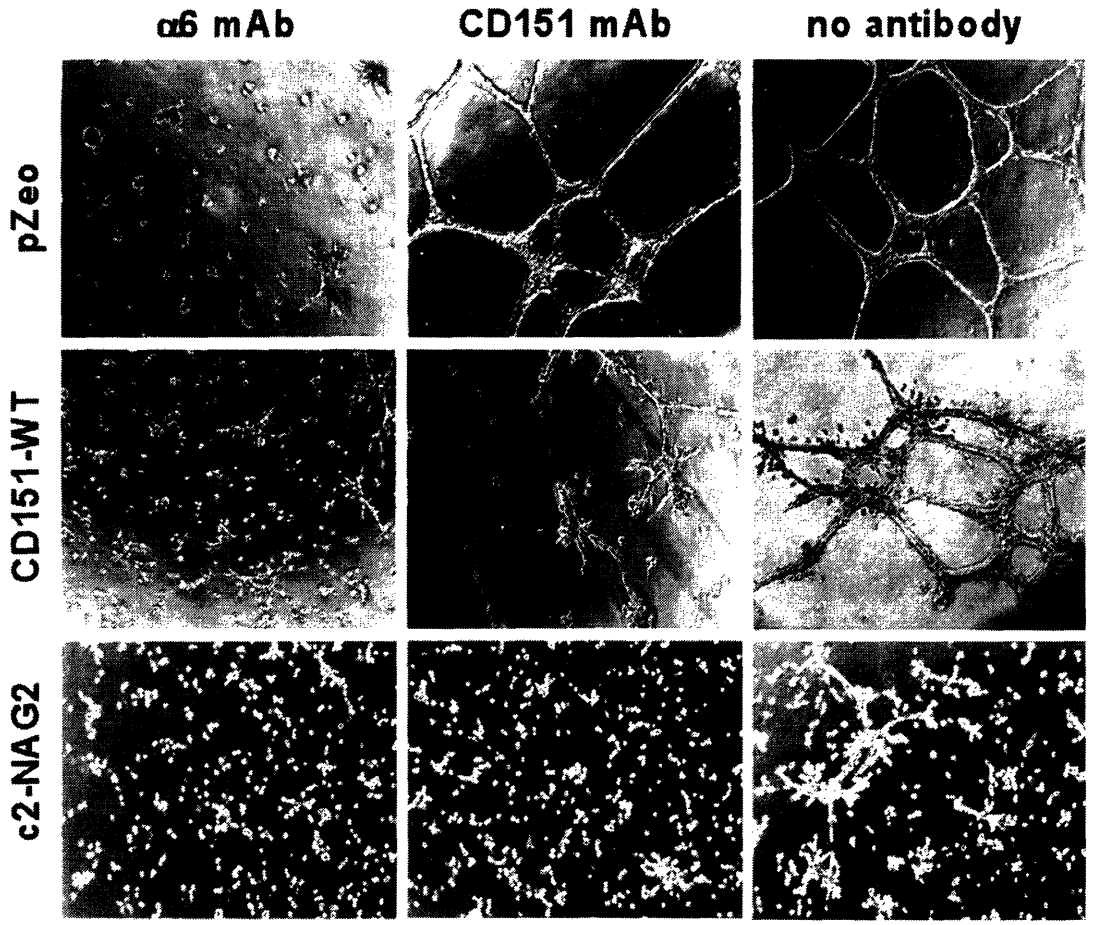


Figure 3.1. CD151-α6β1 integrin complex contributes to NIH3T3 cell assembly into cables. As described previously (20), mock-transfected or CD151-transfected NIH 3T3 cells were plated for 18 h on the surface of a thick layer of Matrigel in 5% FBS-DMEM at 5×10^4 cells per well in a 24-well plate, analyzed using a ZEISS Axiovert 135 microscope, and photographed. In some cases, anti-CD151 mAb 5C11 or anti-α6 mAb GoH3 were added (7.5 μg/ml) at the beginning of the experiment.

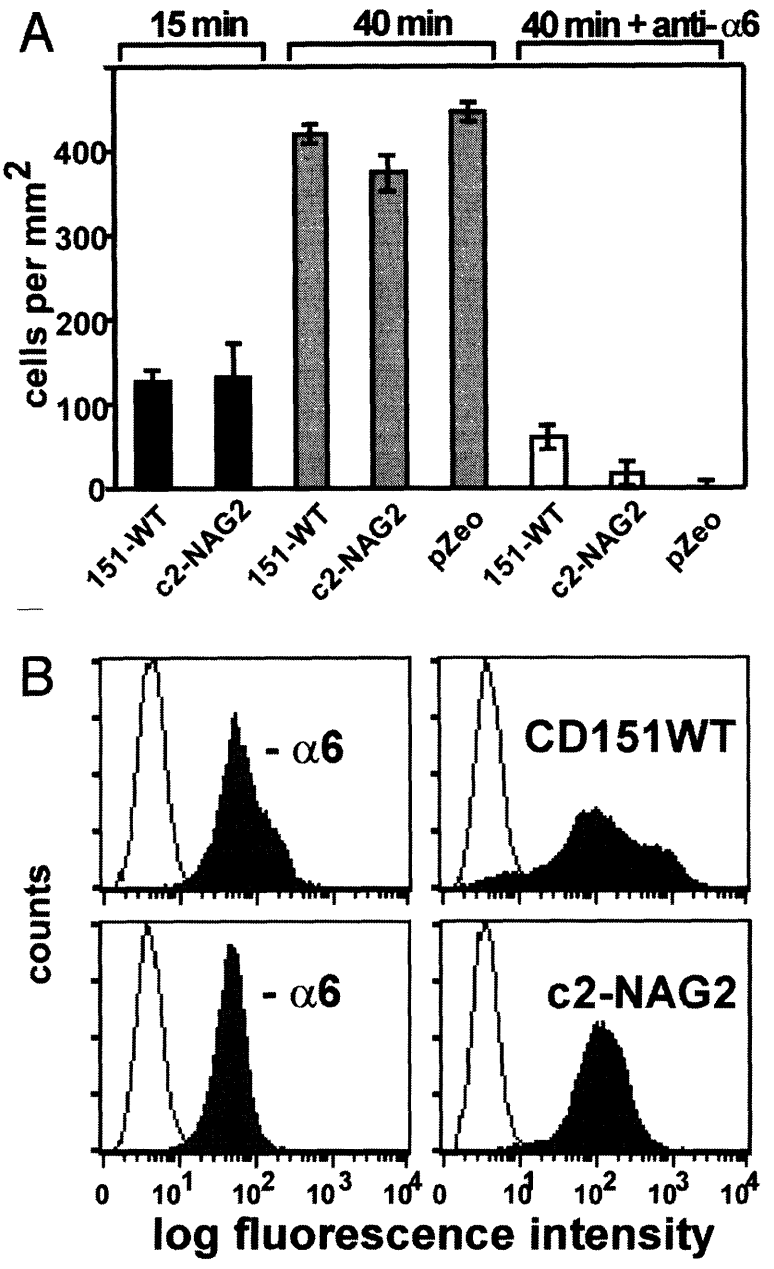


Figure 3.2. Mutant CD151 and CD151-WT effects on static cell adhesion and cell surface expression. (*A*) Transfected NIH 3T3 cells were labeled with BCECF-AM, plated for 15 or 40 min on a plastic surface coated with laminin-1, and after washing, adhesion was quantitated as described (20). In some experiments, anti-α6 mAb GoH3 (7.5 μg/ml) was added. (*B*) NIH 3T3 cells transfected with CD151-WT or CD151-c2-NAG2 were analyzed by flow cytometry, for α6 integrin (mAb GoH3) or CD151 (mAb 5C11) as described (20).

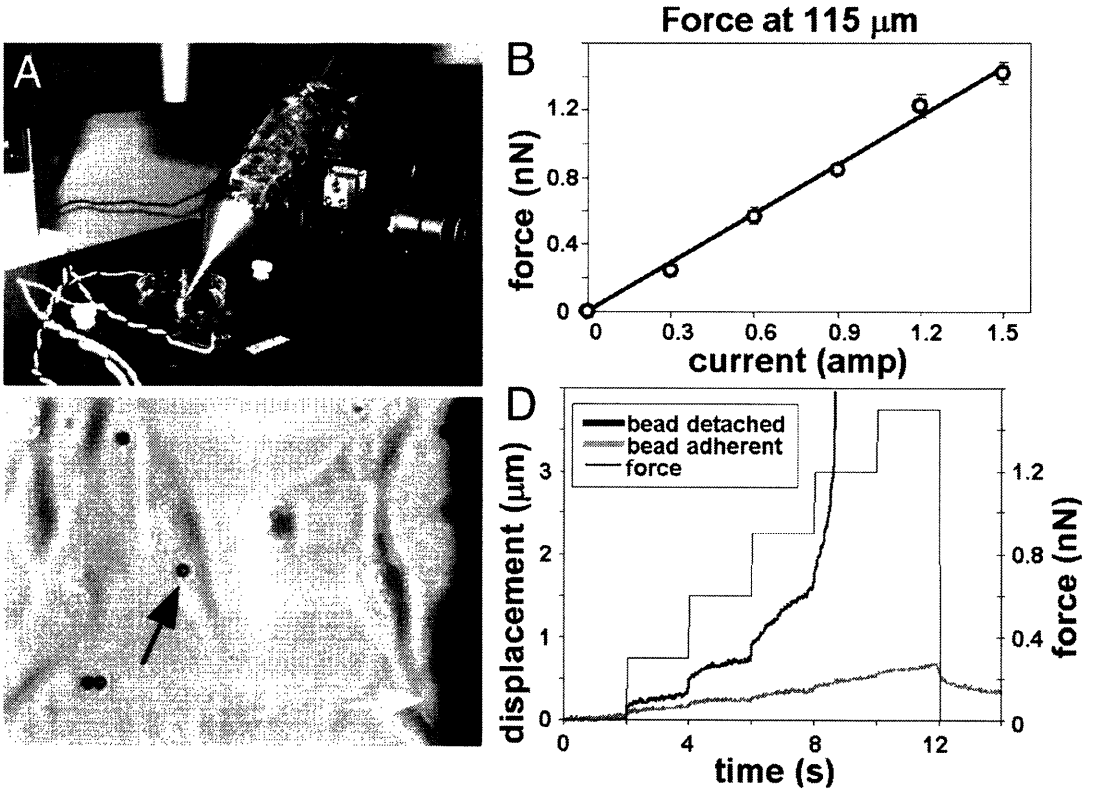


Figure 3.3. Magnetic trap characterization. (*A*) Magnetic trap with temperature controlled stage. (*B*) Shown is force as a function of electric current at a distance of 115 μm from the magnetic trap tip. Line shows linear regression forced through origin after the equation *force*=0.965 nN/amp (R^2 = 0.9948). (*C*) Bright-field image of cell with bead (black arrow) and magnetic trap tip (white arrow), at ×30 magnification.(*D*) Force-displacement profile for two single beads, one detaching at 1.2 nN.

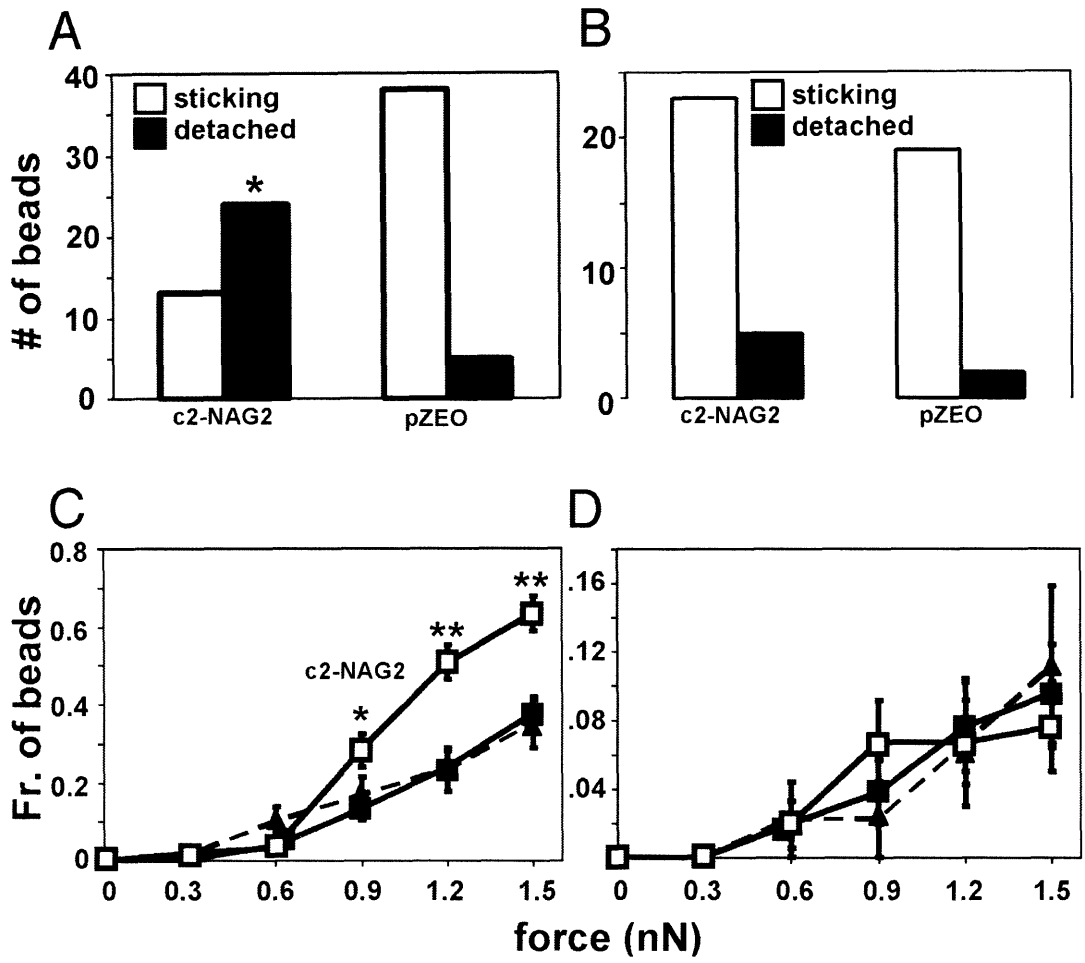


Figure 3.4. Detachment of laminin and fibronectin-coated beads from NIH 3T3 transfectants. (*A*) Detachment of laminin-1-coated beads at 1 nN (*, $P < 0.0001$). (*B*) Detachment of fibronectin-coated beads at 1 nN. (C) bead detachment fraction at increasing force levels for laminin-coated beads (*, $P < 0.01$ vs CD151-WT cells; **, $P < 0.001$). (*D*) Shown is the bead detachment fraction at increasing force levels for fibronectin-coated beads. At lower fibronectin coating levels, we could decrease bead *attachment*, but those beads that attached firmly withstood even the highest forces (>3 nN) without detaching.

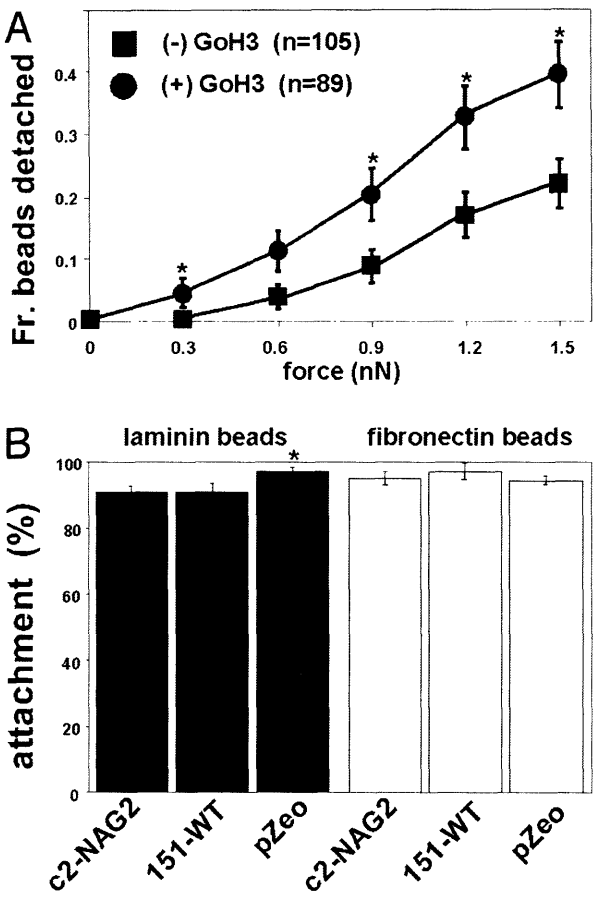


Figure 3.5. Bead detachment effects of anti-α6 antibody and bead attachment by laminin and fibronectin beads. (*A*) NIH 3T3-CD151-WT cells were incubated in the presence (circles, n=89) or absence (squares, n=105) of mAb GoH3 (5-10 μg/ml) at the time of bead attachment (*, $P < 0.05$). The nature of the detachment assay requires selection of beads that have already attached, and hence are more difficult to displace due to antibody inhibition. In contrast, during the static cell adhesion assay, integrin contacts with laminin-1 are not allowed to develop when the inhibitory antibody is present. Hence, GoH3 shows only a partial effect in *A*, compared to a more complete inhibition in Fig. 2*A*. (*B*) For laminin-1-coated beads, attachment fractions = 0.911 ± 0.021, 0.912 ± 0.024 and 0.973 ± 0.013 for CD151-c2-NAG2, CD151-WT, and pZeo, respectively (*, $P < 0.05$ vs. CD151-c2-NAG2 and CD151 WT). (*C*) For fibronectin-coated beads, attachment fractions = 0.951 ± 0.019; 0.973 ± 0.015, and 0.946 ± 0.023 for CD151-c2-NAG2, CD151-WT, and pZeo, respectively.

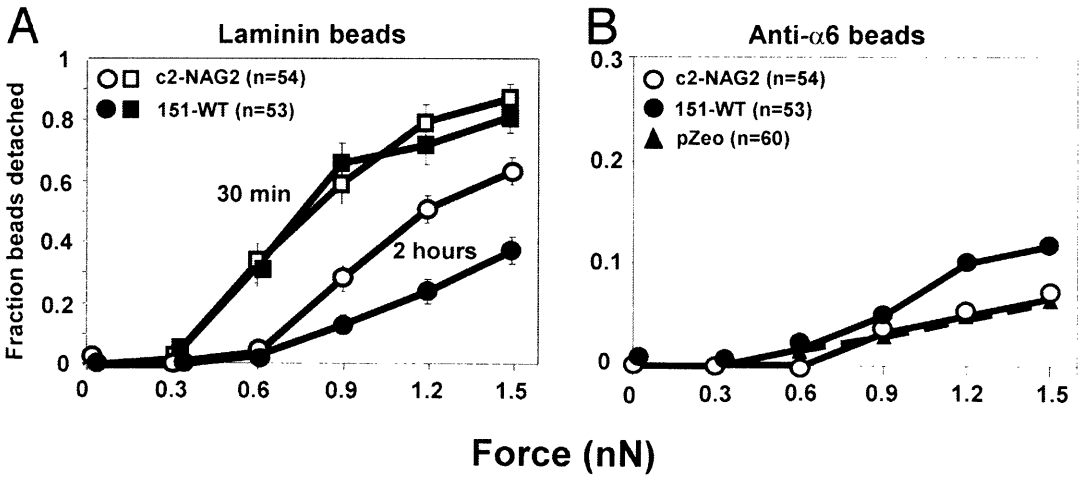


Figure 3.6. CD151 mutant effects are time-dependent and ligand dependent. (*A*) Shown is the bead detachment fraction at increasing force levels for laminin-coated beads after only 30 min of attachment (squares). For comparison, detachment results after 2 h of bead attachment are also shown (circles, as in Fig. 4*C*). (*B*) Shown is the bead detachment fraction at increasing force levels for anti-α6 antibody-coated beads that had been allowed to attach for 2 h.

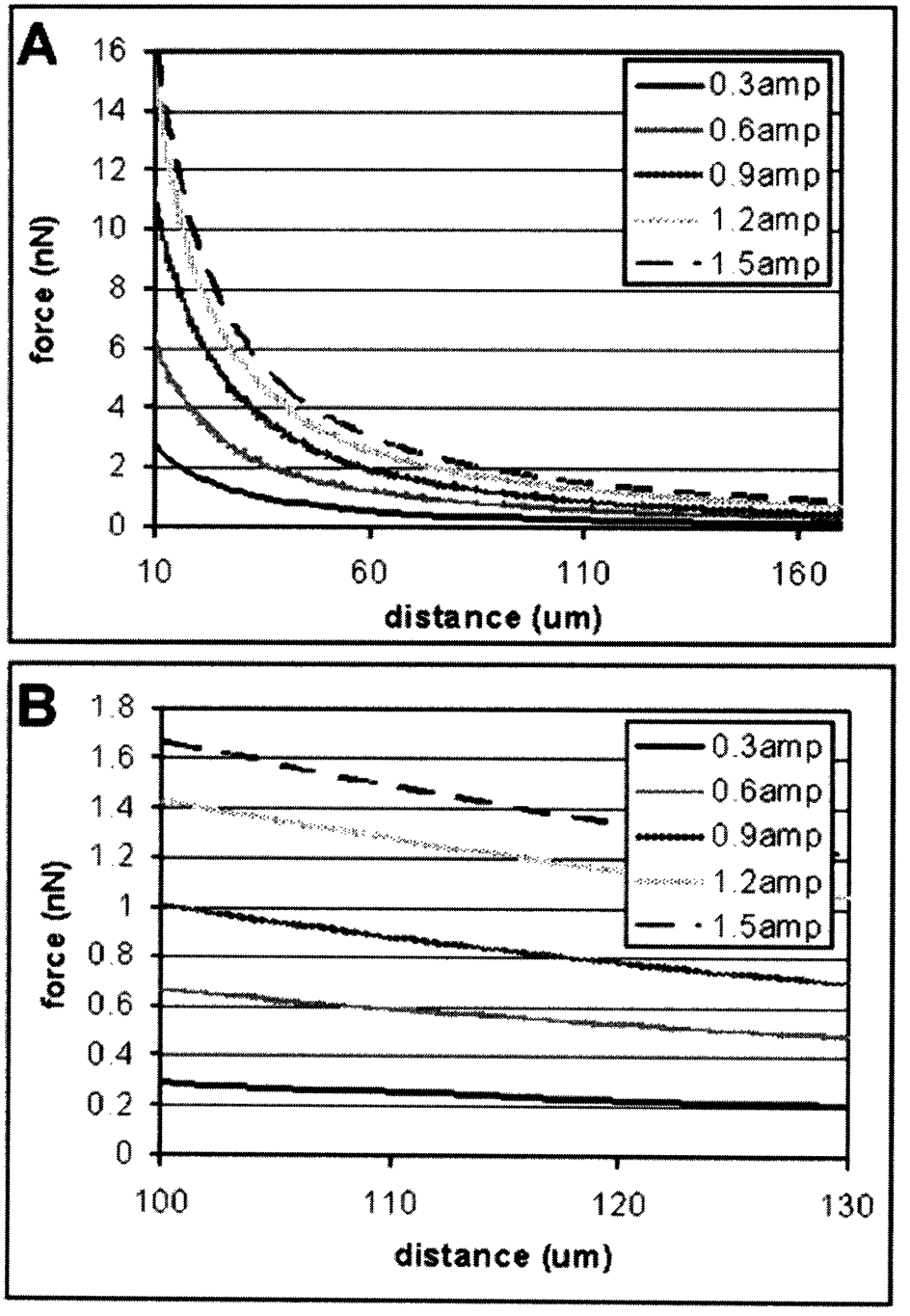


Figure 3.7. Magnetic trap calibration. (*A*) Force-displacement curve over a wide range of distances from the magnetic trap. Curves represent exponential regression based on the least-squares fit to at least three bead experiments at each force level. (*B*) Closeup of distances 100–130 m m from the magnetic trap.

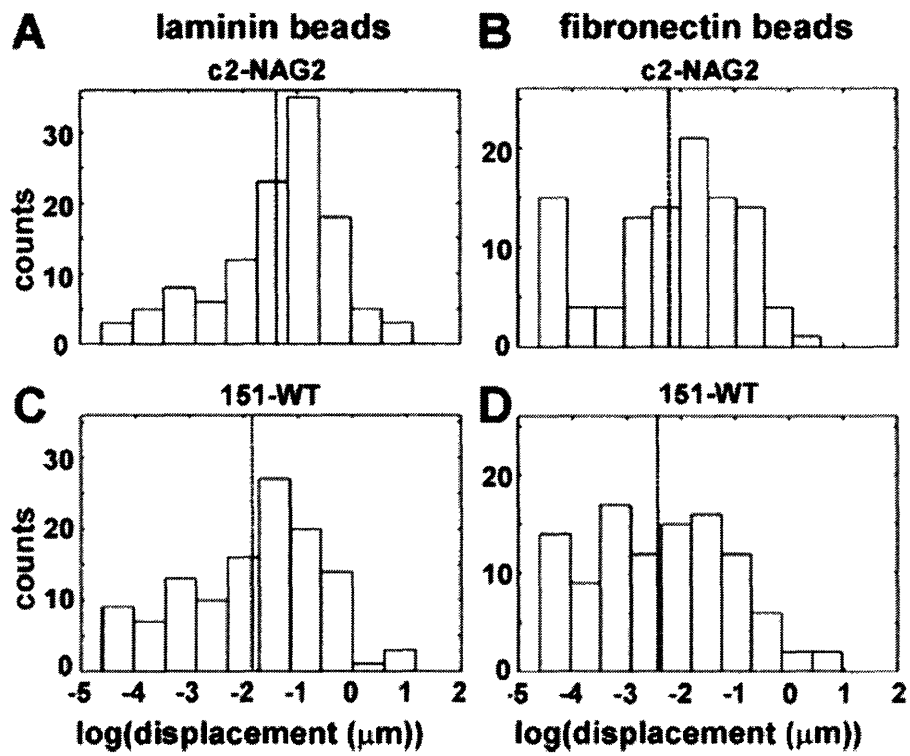


Figure 3.8. Bead displacement for laminin-coated and fibronectin-coated beads. Displacement histograms of laminin-coated beads (*A*) and fibronectin-coated beads (*B*) on CD151-c2-NAG2 cells. Displacement histograms of laminin-coated beads (*C*) and fibronectin-coated beads (*D*) on CD151-WT cells. All histograms are log-transformed to yield Gaussian distributions (the mean is denoted by a dashed line).

3.5 References

1. Ingber, D. (1991) *Curr Opin Cell Biol* **3**, 841-848.
2. Hynes, R. O. (1987) *Cell* **48**, 549-554.
3. Yamada, K. M. & Miyamoto, S. (1995) *Curr. Opin. Cell Biol.* **7**, 681-689.
4. Burridge, K. & Chrzanowska-Wodnicka, M. (1996) *Ann. Rev. Cell Dev. Biol.* **12**, 463-519.
5. Zamir, E. & Geiger, B. (2001) *J Cell Sci* **114**, 3583-3590.
6. Choquet, D., Felsenfeld, D. P. & Sheetz, M. P. (1997) *Cell* **88**, 39-48.
7. Heidemann, S. R., Kaech, S., Buxbaum, R. E. & Matus, A. (1999) *J. Cell Biol.* **145**, 109-122.
8. Wang, N., Butler, J. P. & Ingber, D. E. (1993) *Science* **260**, 1124-1127.
9. Bausch, A. R., Hellerer, U., Essler, M., Aepfelbacher, M. & Sackmann, E. (2001) *Biophys J* **80**, 2649-2657.
10. Gu, J., Sumida, Y., Sanzen, N. & Sekiguchi, K. (2001) *J Biol Chem* **276**, 27090-27097.
11. Attur, M. G., Dave, M. N., Clancy, R. M., Patel, I. R., Abramson, S. B. & Amin, A. R. (2000) *J Immunol* **164**, 2684-2691.
12. de Fougerolles, A. R., Chi-Rosso, G., Bajardi, A., Gotwals, P., Green, C. D. & Koteliansky, V. E. (2000) *Immunity* **13**, 749-758.
13. Hemler, M. E. (1998) *Curr. Opin. Cell Biol.* **10**, 578-585.
14. Boucheix, C. & Rubinstein, E. (2001) *Cell Mol Life Sci* **58**, 1189-1205.
15. Sterk, L. M., Geuijen, C. A., Oomen, L. C., Calafat, J., Janssen, H. & Sonnenberg, A. (2000) *J Cell Biol* **149**, 969-982.
16. Sterk, L. M., Geuijen, C. A., van Den Berg, J. G., Claessen, N., Weening, J. J. & Sonnenberg, A. (2002) *J Cell Sci* **115**, 1161-1173.
17. Yauch, R. L., Berditchevski, F., Harler, M. B., Reichner, J. & Hemler, M. E. (1998) *Mol. Biol. Cell* **9**, 2751-2765.
18. Yánez-Mó, M., Alfranca, A., Cabañas, C., Marazuela, M., Tejedor, R., Ursa, M. A., Ashman, L. K., De Landázuri, M. O. & Sánchez-Madrid, F. (1998) *J. Cell Biol.* **141**, 791-804.
19. Stipp, C. S. & Hemler, M. E. (2000) *J. Cell Sci.* **113**, 1871-1882.
20. Zhang, X. A., Kazarov, A. R., Yang, X., Bontrager, A. L., Stipp, C. S. & Hemler, M. E. (2002) *Mol Biol Cell* **13**, 1-11.
21. Yánez-Mó, M., Mittelbrunn, M. & Sanchez-Madrid, F. (2001) *Microcirculation* **8**, 153-168.
22. Berditchevski, F. (2001) *J Cell Sci* **114**, 4143-4151.
23. Hemler, M. E. (2001) *J Cell Biol* **155**, 1103-1107.
24. Bausch, A. R., Moller, W. & Sackmann, E. (1999) *Biophys J* **76**, 573-579.
25. Bausch, A. R., Ziemann, F., Boulbitch, A. A., Jacobson, K. & Sackmann, E. (1998) *Biophys J* **75**, 2038-2049.
26. Alenghat, F. J., Fabry, B., Tsai, K. Y., Goldmann, W. H. & Ingber, D. E. (2000) *Biochem. Biophys Res Commun.* **277**, 93-99.
27. Meyer, C. J., Alenghat, F. J., Rim, P., Fong, J. H., Fabry, B. & Ingber, D. E. (2000) *Nat Cell Biol* **2**, 666-668.
28. Wang, N. & Ingber, D. E. (1995) *Biochem. Cell Biol* **73**, 327-335.

29. Rychly, J., Pommerenke, H., Durr, F., Schreiber, E. & Nebe, B. (1998) *Cell Biol Int* **22**, 7-12.
30. Huang, H., Kamm, R. D., So, P. T. & Lee, R. T. (2001) *Hypertension* **38**, 1158-1161.
31. Yauch, R. L., Kazarov, A. R., Desai, B., Lee, R. T. & Hemler, M. E. (2000) *J. Biol. Chem.* **275**, 9230-9238.
32. Zhang, X. A. & Hemler, M. E. (1999) *J. Biol. Chem.* **274**, 11-19.
33. Bazzoni, G., Shih, D.-T., Buck, C. A. & Hemler, M. E. (1995) *J. Biol. Chem.* **270**, 25570-25577.
34. Kazarov, A. R., Yang, X., Stipp, C. S., Sehgal, B. & Hemler, M. E. (2002) *J Cell Biol* **158**, 1299-1309.
35. Berditchevski, F., Gilbert, E., Griffiths, M. R., Fitter, S., Ashman, L. & Jenner, S. J. (2001) *J Biol Chem.* **276**, 41165-41174.
36. Davis, G. E. & Camarillo, C. W. (1995) *Exp Cell Res* **216**, 113-123.
37. Vernon, R. B. & Sage, E. H. (1995) *Am J Pathol* **147**, 873-883.
38. Miyamoto, S., Teramoto, H., Coso, O. A., Gutkind, J. S., Burbelo, P. D., Akiyama, S. K. & Yamada, K. M. (1995) *J. Cell Biol.* **131**, 791-805.
39. Lotz, M. M., Burdsal, C. A., Erickson, H. P. & McClay, D. R. (1989) *J. Cell Biol.* **109**, 1795-1805.
40. Garcia, A. J., Huber, F. & Boettiger, D. (1998) *J Biol Chem* **273**, 10988-10993.
41. Lee, I. & Marchant, R. E. (2001) *Surface Science* **491**, 433-4.
42. Lehenkari, P. P. & Horton, M. A. (1999) *Biochem. Biophys Res Commun.* **259**, 645-650.
43. Prechtel, K., Bausch, A. R., Marchi-Artzner, V., Kantlehner, M., Kessler, H. & Merkel, R. (2002) *Phys. Rev Lett* **89**, 028101.
44. Litvinov, R. I., Shuman, H., Bennett, J. S. & Weisel, J. W. (2002) *Proc Natl Acad Sci U S A* **99**, 7426-7431.
45. Shao, J. Y. & Hochmuth, R. M. (1999) *Biophys J* **77**, 587-596.
46. Freyman, T. M., Yannas, I. V., Yokoo, R. & Gibson, L. J. (2001) *Biomaterials* **22**, 2883-2891.
47. Galbraith, C. G. & Sheetz, M. P. (1999) *J Cell Biol* **147**, 1313-1324.
48. Sincock, P. M., Fitter, S., Parton, R. G., Berndt, M. C., Gamble, J. R. & Ashman, L. K. (1999) *J Cell Sci* **112**, 833-844.

4 Experimental verification of a three-dimensional visco-elastic finite element model for cell deformation*

4.1 Introduction

A variety of experimental methods have been developed over the last two decades to probe cellular mechanical properties. These methods include micropipette aspiration, atomic force microscopy (AFM), magnetic twisting cytometry, and magnetic bead microrheolgy and are reviewed in detail in chapter 1. These methods have in common that they apply precisely controlled force or deformation to the cell (surface) and measure induced cellular deformation. In many cases, the cellular deformation can only indirectly inferred from displacements of the force probe (AFM tip, magnetic bead, etc.) that is in close contact with the cells. In order to derive quantitative parameters for cellular mechanical properties such as an apparent elastic modulus from the force and displacement measurements, one has to assume an appropriate mechanical model for the cell material, i.e. a set of constitutive equations that describe the relationship between applied stress and observed strain or vice versa as a function of the specific material properties. In addition to allowing quantitative analysis of cellular material properties, theoretical models of cellular biomechanics can be used to estimate cellular membrane strains and force distribution within the cytoskeleton. Stretch activated ion channels in the cell membrane are activated when forces acting within the lipid bilayer rise to a level sufficient to produce a conformational change in the protein channel and thereby alter its conductance (1). Furthermore, proteins connected to the cytoskeletal network can experience conformational changes in response to forces transmitted through the cytoskeleton, potentially alter their binding affinity to signaling molecules (2, 3).

The mechanical behavior observed in cells is often approximated by isotropic, linear, elastic material properties, and cell and probe geometry are simplified so that the problem can be analytically solved, e.g. in the wide spread use of a Hertz model for AFM

* This chapter contains sections of an article that has been published in the *Biophysical Journal* 2003, vol. 85: 3336-3349.

indentation measurements. More complex models have been developed that better reflect the anisotropic microstructure of the cytoskeletal network, including the Open-Foam model by Satcher and Dewey (4) or the tensegrity model by Ingber (5, 6). Theoretical estimates derived from these models more or less agree with values observed in micromanipulation experiments (7), but these theoretical models fail to account for the viscoelastic component that is observed in many cell deformation experiments (8).

Here, we experimentally verify a computational finite element model developed by Helene Karcher (9) that simulates forcing of a paramagnetic microbead on a single large cell or continuous cell monolayer and predicts surface and internal mechanical stress/strain distributions. The three-dimensional model incorporates viscoelastic properties for the cytoskeleton and membrane/cortex composite, and allows for modulation of cell height and material properties to investigate the behavior of different cell types under mechanical stimulation. Model predictions are compared to experimental results obtained with time-varying force to assess model validity. Predictions derived from the model can then be used to: (*i*) determine the mechanical properties of the cells by comparison to experimental results, (*ii*) correlate the localized stress/strain patterns to biological responses of the cell, and (*iii*) provide validation for a simple theoretical model that can be used to interpret other experimental observations.

4.2 Materials and methods

Magnetic trap calibration. The magnetic trap was calibrated by suspending magnetic beads (Dynal, Dynabeads M-450) in Dimethylpolysiloxane (Sigma, DMPS – 12M) and tracking the position of the beads as they are attracted to the magnetic trap over a range of electrical currents (0.3 to 1.5 amps). Details on the magnetic trap design and operation are provided in (10).

Bead coating with extracellular matrix proteins. Magnetic beads were coated with fibronectin (GibcoBRL, 33016-023) according to the manufacturer's protocol with the following modifications: fibronectin was applied at a final concentration of 500 μg/mL in borate buffer (pH 8.5) for 18 hours at 37°C.

Cell culture. NIH 3T3 fibroblasts were maintained in Dulbecco's modified Eagle's medium (DMEM, Whittaker) supplemented with 10% fetal calf serum and antibiotics.

Experimental procedure. Polystyrene cell culture dishes (Corning) were coated with 0.1% gelatin in phosphate buffered saline (PBS, Gibco) over night at 4°C to facilitate cell attachment. Cells were plated in DMEM supplemented with 10% fetal calf serum (FCS), penicillin/streptomycin and Zeocin (200 μg/mL) at 3 mL/dish on the gelatin coated dishes at a density of 150,000 cells/dish and incubated at 37°C overnight. Medium was replaced the next day with DMEM containing 5% FCS and 6 μL fibronectin coated bead suspension (final concentration 1.2×10^6 beads/dish) and incubated at 37°C for 45 min to guarantee sufficient bead attachment to the cells. The cell culture dish was then placed on a temperature-controlled stage. Cells with adherent beads were imaged at ×30 magnification using an inverted light microscope (Olympus, IX-70). Non-confluent cells with a single bead firmly attached to the flat section of the cell surface were selected for the magnetic trap experiments. The magnetic trap was brought into a parfocal position with the bead at a distance of 115 μm away from the magnetic trap tip. One of the following force profiles was then applied to the bead while recording the bead position with a digital camera (Roper Megaplus ES310/T) at 60 frames per sec:

1. Sine wave: a force free period of one second followed by 8 sec of a 1 Hz sine-wave pattern with amplitudes of 0.125 nN or 0.6 nN and an offset of one amplitude, followed by 1 s at zero force to monitor the relaxation of the bead.
2. Step function: a constant force rising in steps of 300 pN every 2 sec, so that it reaches 1500 pN after 5 steps
3. Force ramp: linearly increasing force from 0 to 500 pN at a rate of 250 pN/sec, followed by 2 sec at zero force.

A Gaussmeter probe (LakeShore) was used to simultaneously measure the magnetic field during the force application, and the read-out was saved with the video image data. Subsequent cells were selected at least 5 mm away from any previous force application sites to avoid studying pre-conditioned cells. Five to fifteen cells were selected in each dish, and the experiments were concluded within 30 min per dish.

Particle tacking and phase lag determination. Custom-written MATLAB (MathWorks, Natick, MA) software that uses a combination of cross-correlation and center-of-mass computation was used to track the bead centroid position from the digitally recorded videos with a spatial resolution of ~10 nm at ×30 magnification. The phase lag between the applied force, represented by the magnetic field strength, and the resulting bead displacement was computed using cross-correlation analysis. The temporal resolution is one frame, i.e. 1/60 sec.

The validity of the phase lag measurements was confirmed by applying the technique to beads embedded in purely viscous (dimethylpolysiloxane) or purely elastic media (polyacrylamide gel). Bead displacements in the elastic media exhibited a negligible phase shift of 2.95° ± 3.564°, while displacements of beads in viscous media showed a phase lag close to 90°: 86.91° ± 5.94°, i.e. force corresponded to the derivative of the displacement.

For the force step function, maximal displacement was defined as the difference in mean bead position between the last 10 frames (= 0.4 sec) of force application at each force level and the initial position, estimated as the mean bead position during the 25 frames (= 1 sec) prior to the application of force.

Computational model. Details on model geometry, boundary conditions, mechanical and material properties, sensitivity analysis and solution techniques can be found at Karcher et al. (9). In brief, the models consists of a cylindrical domain (Fig. 4.1) representing a portion of a continuous cellular monolayer or a large, spread out cell. The discrete nature of cytoskeletal filament network – microtubules, actin and intermediate filaments – was not depicted, based on the observation that the relevant length scale present in the application of force via a tethered bead is considerably larger than the

filament network mesh size (~50-100 nm). The lateral extent of the monolayer was chosen large enough (40 μm) to eliminate any effect of the boundary on the stress or strain distributions in the vicinity of the bead, and a zero-displacement boundary condition was imposed for the bottom surface. A reference model with a 10-μm high and 40-μm wide cylindrical monolayer was implemented, and its height was modulated to depict different cell types. The cell monolayer consists of two parts: (*i*) the cytoskeleton, *i.e.* the main part of the cylinder, and (*ii*) the membrane and the actin cortex, a shell layer atop cytoskeleton. The membrane and the cytoskeleton were represented by either a "fluid-like" viscoelastic Maxwell model analogous to a spring and a dashpot in series or a "solid-like" Voigt model analogous to a spring and a dashpot in parallel. The baseline viscoelastic parameters of the cytoskeleton were chosen as G_c = 100 Pa and μ_c = 100 Pa·s, yielding a characteristic time (μ_c / G_c) of 1 sec for the viscoelastic behavior. When the cytoskeletal properties are varied (200, 400, 600 and 1000 Pa) in the results presented below, this characteristic time is assumed constant. The membrane/cortex layer was modeled as an incompressible shell of constant thickness with a bending stiffness of K_b = 2 x 10^{-19} to 2 x 10^{-18} N·m and a viscoelastic time constant of τ = 0.005 to 0.1 sec. The adherent paramagnetic bead was modeled as a rigid body, and the bead immersion half-angle was chosen as α = 45°, corresponding to a contact radius of 1.6 μm (Fig. 4.1). To probe the effect of varying degrees of bead contact, simulations were also performed at half angles of α = 60° and 75°.

4.3 Results

4.3.1 Numerical model

Model dependence of the results

Simulations with either the Maxwell or Voigt models exhibited similar deformation and stress distributions, but required different values for the shear modulus to best fit the experimental data. The best fit value for the shear modulus was 600 Pa for the Maxwell model and 100 Pa for the Voigt model, both using a time constant of 1 sec. In separate experiments in which a step-wise force was applied to the bead and held for 4 sec, the bead invariably immediately displaced, and then continued to creep (Fig. 4.2). In some

cases, the creep continued at nearly a constant rate, suggestive of a Maxwell model, while in others it approached a constant asymptotic value. Because the Maxwell model seemed somewhat more consistent with experimental observations, all the following comparisons use the Maxwell description in all subsequent simulations.

Bead behavior

Magnetic forcing produced both translation in the x-direction and rolling about the y-axis (due to the induced torque around the bottom of the bead fixed to the cell) – see Fig. 4.3. After 2.0 sec, the bead center translation was 1.67 μm and the bead had rolled approximately $\theta \sim 20$ degrees (Figs. 4.3). This means that, due to the rotation, the cell surface over the region attached to the bead was displaced less than the bead center, by an amount equal to $R\theta \sim 0.8$ μm in this case, where R is the radius of the bead.

The viscoelastic response of the monolayer was evident when the bead was forced sinusoidally. Because the forcing time scale is comparable to the cytoskeleton relaxation time scale and much larger than the viscoelastic time constant of the membrane/cortex, the force-displacement curve is effectively dominated by the characteristics of the cytoskeleton. This was confirmed by simulations with the membrane/cortex shell removed, for which the bead force-displacement relationship is virtually identical to the complete simulation including the membrane/cortex.

The overall character of the force-displacement curves compare favorably with our measurements performed on NIH 3T3 fibroblasts (Fig. 4.5). Firstly, the experimental and numerical curves both exhibit the same convex shape, characteristic for viscoelastic materials. Secondly, the simulation agrees most closely with the experimental data when the cytoskeleton shear modulus are set between 600 Pa and 1 kPa, consistent with reported values for the shear modulus of ~1 kPa for chick fibroblasts (Thoumine and Ott, 1997). Increasing the cytoskeleton shear modulus did not change the overall trend of the bead force-displacement curve, but significantly decreased the bead displacement for the same force applied (Fig. 4.5) (see *Linear displacement studies* below).

General cell monolayer movement

Monolayer movement appeared to be highly localized in the vicinity of the bead (Figs. 4.3 and 4.4), consistent with measurements reported in Huang et al. (10) and Bausch et al. (8). Two distinct regions of large displacement were apparent in all simulations, one in front of the bead and one behind it, so that the overall displacement field exhibits a pulling/squeezing pattern (Fig. 4.3). Interestingly, a zero-displacement zone is visible immediately beneath the bead. Consequently, the forces inside the monolayer are expected to be concentrated ahead of and behind the bead and somewhat diminished directly below it.

In all simulations, the maximum displacement inside the monolayer is in the direction of forcing, located on the membrane, immediately behind the bead, and roughly equal to half of the bead displacement. For example, after 2 sec, i.e. when the force applied is 500 pN, the maximum displacement in the monolayer is 1.02 μm in the forcing direction, whereas the bead center displacement is 1.67 μm. The maximum displacement is smaller in other directions: 0.91 μm and 0.21 μm in the vertical and transverse directions, respectively. Predicted displacements are consistent with measured membrane displacements of around 0.1 μm after 0.2 μm bead displacement a few micrometers away from the bead center (8), and overall displacements under 1 μm (10) for forces of 200 pN.

Influence of the forcing time dependence

Linear displacement studies. Simulations with monolayers having different cytoskeleton shear moduli led to bead displacements scaling approximately with the inverse of the shear modulus provided the time constant of the material was held fixed (Fig. 4.5). For example, after 2 sec, when the force acting on the bead is 500 pN, bead displacement was 0.86 μm for a cytoskeleton shear modulus of 200 Pa but only 0.46 μm when the shear modulus was increased by a factor of two, to 400 Pa.

Application of a sinusoidal force. Bead displacement under sinusoidal forcing exhibits an oscillatory behavior with a time-varying mean (Fig. 4.6), indicative of the viscous

character of the Maxwell model. For the particular conditions of Fig. 4.6 with a forcing frequency of 1 Hz and force amplitude of 125 pN, the net bead displacement per cycle was approximately 0.21 μm along the forcing direction and the phase lag between the bead displacement and the applied force was 0.05 ± 0.005 sec (mean ± standard deviation). Aside from the shift in bead position, the maximum bead displacement decreases slightly from one cycle to the next, from 0.57 μm in the first cycle to 0.50 μm in the fourth cycle.

The displacement and stress patterns in the monolayer (data not shown) are similar to those observed in the simulation conducted with a ramp force applied on the bead as shown above.

4.3.2 Cell experiments

Linear displacement studies. To experimentally test the linearity of the force displacement curves when the force is time-independent, we applied a stepwise increasing force to the beads. The force was increased by 300 pN every 2 sec until a maximal force of 1500 pN was reached. For the analysis, only beads that remained attached throughout the entire force application were considered (96 out of 104). Not only was the mean displacement versus force relationship linear, but also almost all beads exhibit a linear force-displacement relationship over the entire range of forces. The linear regression gave

$$\textit{bead displacement } (\mu m) = a + b\,[\textit{force } (nN)] \qquad (1)$$

where a = -0.045 μm, with a 95% confidence interval between -0.299 and 0.209 μm, i.e. no significant deviation from the expected value of $a = 0$; and b = 0.650 μm/nN, with a 95% confidence interval between 0.394 and 0.905 μm/nN. The departure from linearity in the data was non-significant ($P = 0.9993$) and validated the choice of linear elements to model the cell monolayer material during magnetic bead microrheometry. Linear behavior was also observed in experimental sinusoidal forcing, as discussed below.

Application of a sinusoidal force. At 125 pN force amplitude, less than half the cells (8 out of 19) exhibited a detectable response to the force application (Fig. 4.6). In contrast, almost all cells (18 out of 19) showed a detectable response at 600 pN. Only cells with detectable sinusoidal displacement pattern were selected for calculation of the phase lag between the displacement and the force. For the lower amplitude (125 pN), the phase lag was 0.062 ± 0.041s, in agreement with the numerical model finding of 0.05 s and corresponding to a phase angle of 22.5°. At larger amplitudes (600 pN), the phase lag was 0.068 ± 0.040 s, not significantly different than at low amplitudes (P = 0.7783). This lag indicated a significant viscous component in the mechanical cell response. Considering all 19 beads for both experiments, the displacement amplitude at the low force level (125 pN) was 0.032 ±0.043 μm, compared to 0.118 ±0.123 μm at higher forces (600 pN, still applied at 1 Hz). As indicated by the large standard deviation of the displacement amplitudes, the cells exhibit a highly heterogeneous mechanical response due likely to variations in cellular stiffness and/or contact angle of the bead.

The mean displacements, defined as the total bead displacement averaged over one forcing period, increased with time. This viscous creep is consistent with the computational results (Fig. 4.6). Two distinct patterns were observed in the bead displacement plots versus time. Namely, beads with low displacement amplitudes seem to maintain a rather constant mean displacement over time, while beads with larger displacement amplitudes show an increasing mean displacement over time. These last two observations were consistent in both sets of experiments performed at 125 pN and 600 pN force amplitude.

Statistical Analysis. Multiple regression analysis was used to examine the correlation of phase lag, amplitude and residual bead displacement (defined as the bead displacement after 1 s of relaxation following the 8-cycle sinusoidal force application) for the 600 pN amplitude experiments. The statistical analysis was performed with 18 cells out of 19, excluding one cell with displacement amplitude lower than the resolution limit (10 nm). The residual displacement correlated with the phase lag and the displacement amplitude following a linear model,

$$\textit{Residual displacement } (\mu m) = c + d\,[\textit{phase } (s)] + e\,[\textit{amplitude } (\mu m)] \qquad (2)$$

Where c = -0.217 μm, with a 95% confidence interval between -0.330 and -0.103 μm; d = 4.627 μm/s, with a 95% confidence interval between 3.404 and 5.849 μm/s; e = 1.676, with a 95% confidence interval between 1.238 and 2.115. The correlation was significant: $R^2 = 86.71\%$, with $P < 0.001$ for c and $P < 0.0001$ for d and e.

4.4 Discussion

The predictions for the magnetic bead displacements in response to linear and sinusoidal forcing obtained from the finite element model are in excellent agreement with our experimental data (see figs. 4.5 and 4.6). The best fit values for the shear module (~600 Pa for the Maxwell model using a time constant of 1 sec) are also consistent with measurements obtained from other groups that predict apparent elastic moduli and shear moduli on the order of 1000 Pa using magnetic bead microrheology (8, 11), microplate indentation (12), atomic force microscopy (13), and micropipette aspiration (14). In contrast, other groups have reported values for cytoskeletal stiffness as low as ~10-100 Pa and as high as 100 kPa, with the lower values generally obtained from magnetic twisting cytometry and the higher values derived from AFM experiments. These apparent differences of up to 5 orders of magnitude can only in part be attributed to cell type specific differences or the very heterogeneous cellular mechanical response (cell-to-cell and intracellular variations) and most likely reflect differences in applied force application and experimental analysis methods. A detailed analysis by Fabry et al. (15) suggests that heterogeneous bead attachment has a large effect on apparent mechanical

properties measured by magnetic twisting cytometry. A relatively small fraction of unbound beads can lead to significant underestimation of cell stiffness, especially when low magnetic torque is used. Their calculations reveal that cells with an apparent stiffness of 200 Pa and a 10% fraction of unbound beads would yield a measured apparent stiffness of 25 Pa when stressed with ~1.5 Pa (15). On the other hand, some of the highest values for cytoskeletal stiffness have been reported in AFM measurements, especially on lamellipodia regions and at stress fibers in cells. Here, it is important to consider that atomic force microscopy measurements often apply only small indentations that most likely probe the material properties of the membrane and actin cortex and stress fibers near the cell surface. Furthermore, extremely high stiffness values reported for lamellipodia and other thin cellular regions might overestimate the mechanical properties, since the Hertz assumption of a thick (semi-infinite) medium compressed by a rigid probe is no longer valid and the stiff underlying substrate influences the measured results. Mahaffy et al. (16) have recently applied an improved theoretical model to measurements in thin cellular regions and obtained values for the apparent elastic modulus of ~1.6 ± 0.2 kPa for thin regions and of ~0.6 ± 0.1 kPa for other regions, very close to the values obtained from our model.

However, it is important to notice that this is merely a descriptive model that provides little insight into the underlying cellular and molecular mechanisms that lead to the observed mechanical response. Nonetheless, the model is valuable to estimate the stress and strain distribution on the cell surface and within the cells under conditions similar to the model assumptions, i.e. force applied through micrometer sized beads on a flat cell surface with time-scales on the order of 1 sec. The displacement distribution obtained from the computation model reveals that the bead undergoes significant rotation in addition to translation (see fig. 4.3). Therefore, the center of the bead exhibits significantly larger displacements compared to the membrane and cytoskeleton, and direct inference of membrane displacements from bead displacement would lead to considerable error and consequent underestimation of cytoskeletal stiffness when used in connection with theories that relate surface displacements to deformations and stress within the cell interior (17, 18). This is further compounded by the fact that bead rotation

and membrane displacements strongly depend on the contact angle between bead and cell, and in experiments cells display varies degrees of bead embedding with half-contact angles between the bead and the cell ranging from a 36° to 86°, with a mean a value of 67° (19).

The computed membrane displacements (fig. 4.4) are consistent with previous observation of a localized mechanical response with rapid decay of displacements away from the force application site (8, 10, 20, 21). Even though the current model cannot predict membrane strains since it assumes an incompressible shell with constant thickness, these displacements are likely to be sufficient to activate stretch-activated ion channels and other strain sensitive proteins in the membrane and cytoskeleton. This theory is consistent with observations of contact sensitive calcium induction in chondrocytes probed with an AFM tip (13) and the finding that many adult cardiac mouse myocytes exhibit spontaneous contraction when forcing laminin-coated magnetic beads adherent to the cell surface (Jan Lammerding, unpublished observation).

Interestingly, the simulations also reveal that the cytoskeleton dominates the mechanical response in magnetic bead microrheolgy with forcing functions on the order of 1 sec, and the membrane/cortical layer simply serves to transmit the applied force to the cytoskeleton. Computations conducted without the membrane/shell cortex produced virtually identical results. However, some of these observations could also be attributed to the specific selection of finite element model elements and material properties for the membrane/cortical shell and need further experimental verification.

Our experimental data confirms that the cytoskeleton exhibits a viscoelastic response with a time constant of ~1 sec. It is important to notice that these observations are limited to similar experimental conditions, as significantly higher (>> 1 Hz) or lower (<< 1 Hz) forcing frequencies could results in different material behavior, as was recently demonstrated by Fabry et al. (22). Due to numerical limitations, our finite element model could simulate only continuously varying forcing functions, i.e. sine waves and force ramps. For these forcing functions, both Voigt and Maxwell model resulted in

indistinguishable displacement curves, but resulted in different estimates for the material property parameters based on the best fit values. Considering that only the Maxwell model would exhibit an immediate displacement following the onset of forcing, it seems a better fit to the experimental step function data. However, both model choices are limited by a combination of only two spring/dashpot elements, and more complex viscoelastic models might capture the observed data even better.

Two other experimental observations are noteworthy. (*i*) Both the sinusoidal forcing at two different force magnitudes and the stepwise increase in force revealed a linear force-displacement relationship. This observation at least in parts justifies the assumption of linear elastic materials used in many current cell mechanics models. However, it is important to notice that this behavior might be limited to the time and force scales used in our experiments. At shorter time scales, viscous effects might dominate over the elastic response, and cytoskeletal remodeling and adhesion strengthening have to be considered for larger time scales. Furthermore, experimental observations reveal that increased forces can lead to the rupture of bonds between cell and bead with subsequent partial or complete bead detachment, or they can result in membrane tethering of the bead, resulting in large deviations from linearity. (*ii*) The statistically significant correlation between residual displacement magnitude and both displacement amplitude and phase lag (see equation 2), provides a quantitative relationship between the plastic deformation (residual bead displacement) and the viscous (phase lag) and elastic (major component of displacement amplitude) material properties and could be used to validate other viscoelastic models. However, other sources for plastic deformations such as partial bead detachment etc. will have to be considered as well.

In conclusion, the finite element model captures most of the experimental aspects of magnetic bead microrheology and allows estimation of cellular mechanical properties as well as prediction of stress and strain distribution within the cells. The specific choice of a Maxwell model and the good overall fit between numerical and experimental data should not suggest that this is the only valid model to describe cellular mechanics. In

contrast, alternative and possibly more complex material models that capture the diverse aspects of the cellular mechanical response more accurately or that better reflect the actual cellular microstructure (cytoskeletal network, nucleus, etc.) should be considered in future work. The advantages of a finite element model over other analytical approaches are that the model can easily be adapted to changes in cell geometry to better reflect experimental conditions and that various material models can implemented into the software, enabling the comparison between specific material models and experimental results. An optimized model of cellular mechanics can then be used to quantitatively analyze cellular mechanical properties and to provide estimates for the intracellular and membrane stress and strain distribution during these experiments.

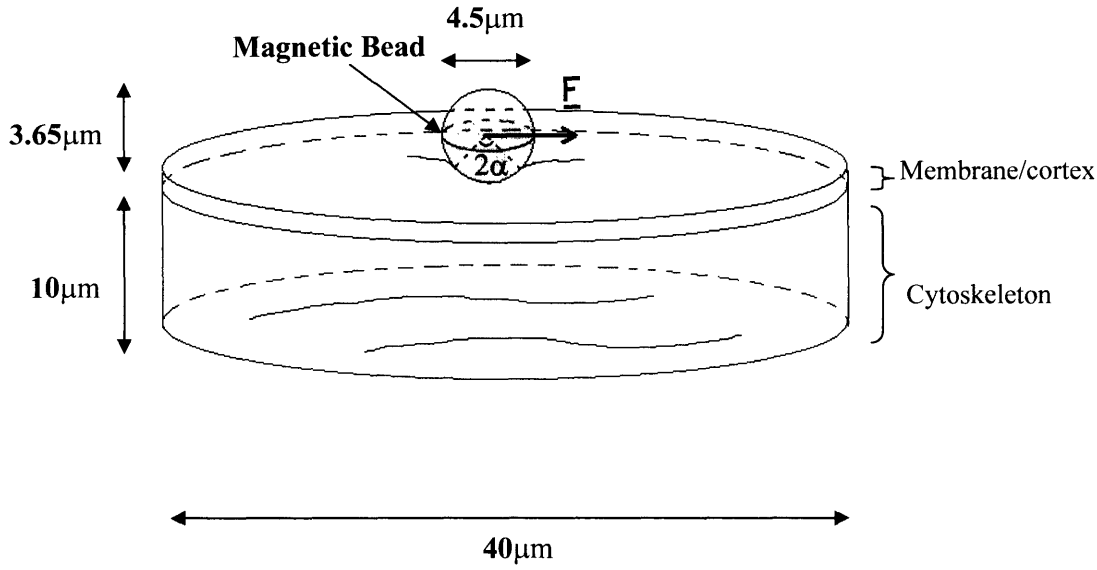


Figure 4.1. General model geometry. The cell monolayer is divided into cytoskeleton and membrane/cortex, each of which are assigned different material properties. All elements are drawn to scale, except for the membrane, the thickness of which is exaggerated for clarity. The contact angle between the bead and the cell monolayer is set to α=90°, so that only 3.65μm of the bead extends from the monolayer.

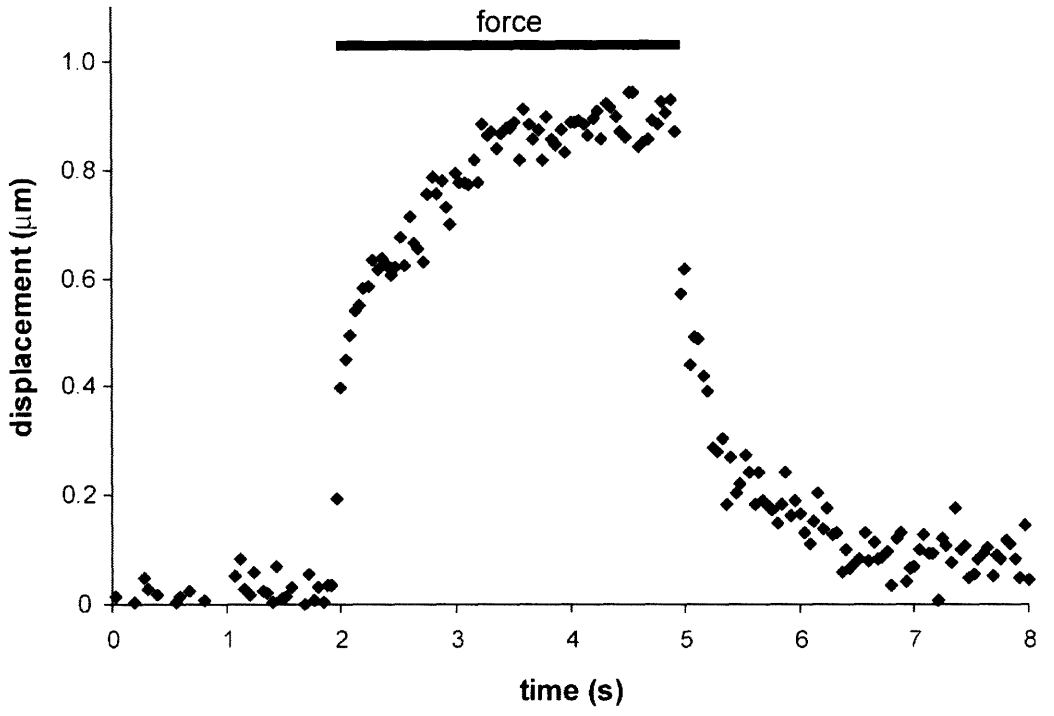


Figure 4.2. Representative magnetic bead displacement for step force application. A force of 1 nN was applied for 3 sec starting at t = 2 sec (see black bar). The displacement plot indicates an immediate elastic response combined with a slower viscous component, resulting in the characteristic viscoelastic response curve.

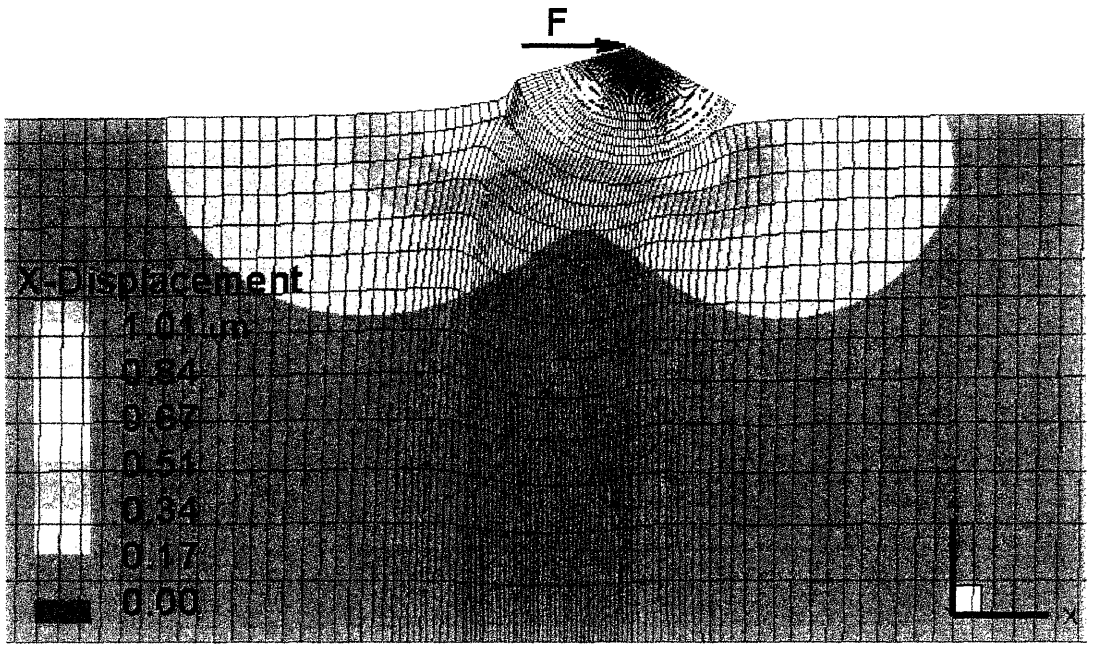


Figure 4.3. Cross-sectional view of monolayer (y=0). Displacement field in the forcing (x) direction after 2.0s. Arrow indicates the force F=500pN applied. Only part of the bead (unfilled gray network) is displayed. The displacement field is localized near the bead and exhibits a pulling/squeezing pattern. No significant displacements occur immediately beneath the bead.

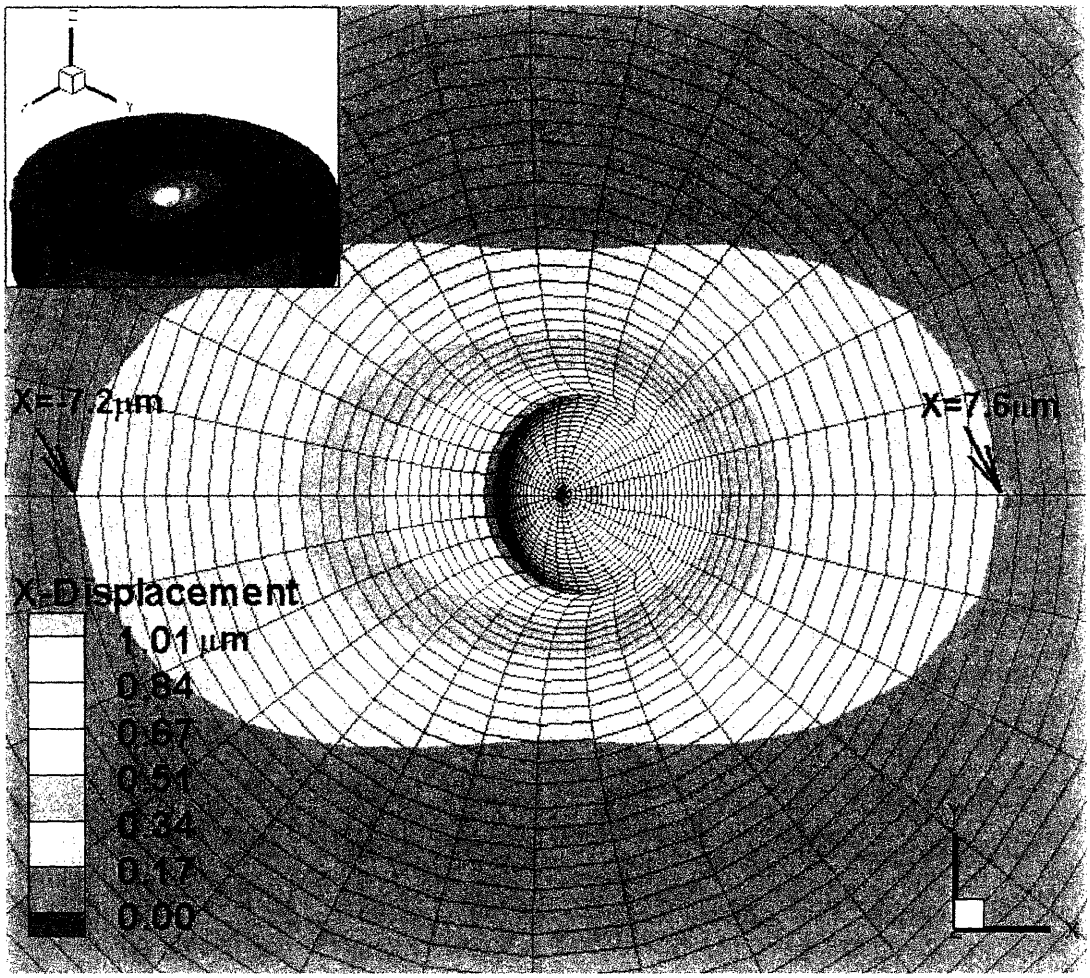


Figure 4.4. Membrane displacement in the forcing direction. Enlarged top view after 2.0s forcing (500pN). Inset shows the displacement field for the whole model and the region of the enlargement (black rectangle). Bead not shown. Displacements are seen to extend more in the forcing (x) direction than in the transverse (y) direction.

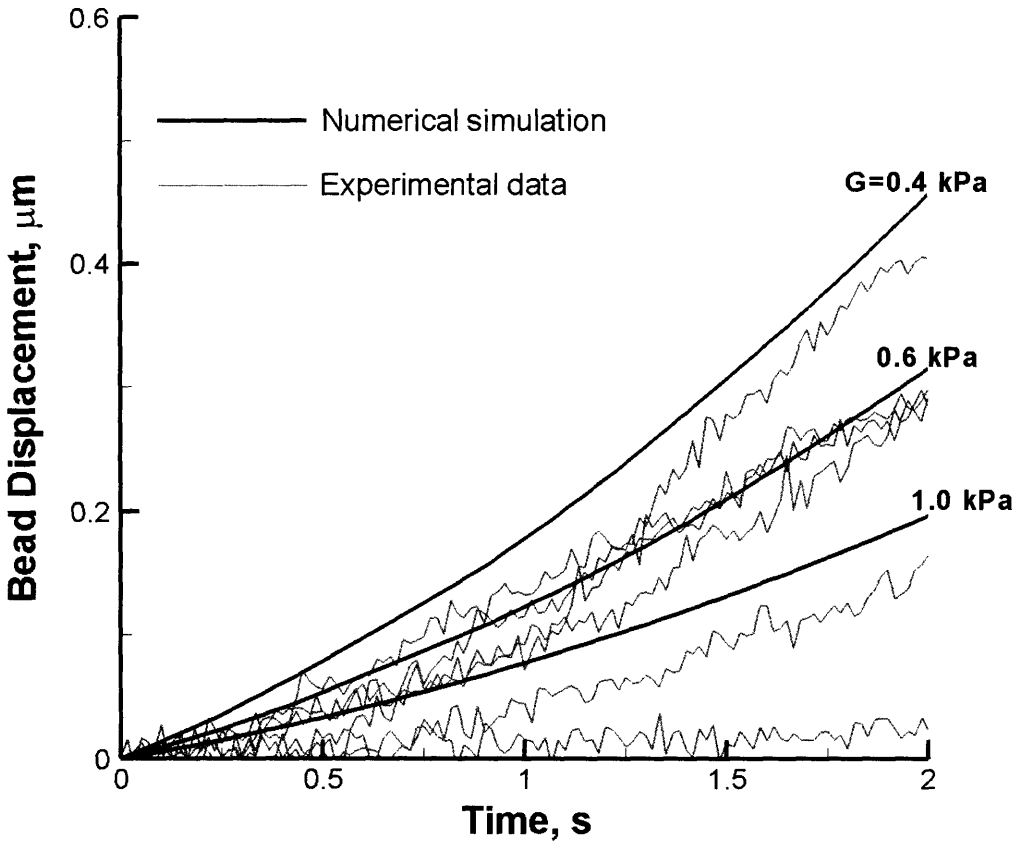


Figure 4.5. Bead center displacement versus time. Numerical results (black curves) are shown for three cytoskeleton shear moduli. The seven gray lines are sample data, each obtained from a different NIH 3T3 fibroblast (see Cell Experiments) within a single experiment. Both numerical and experimental curves were obtained with a force imposed at a constant increasing rate of 250pN/s.

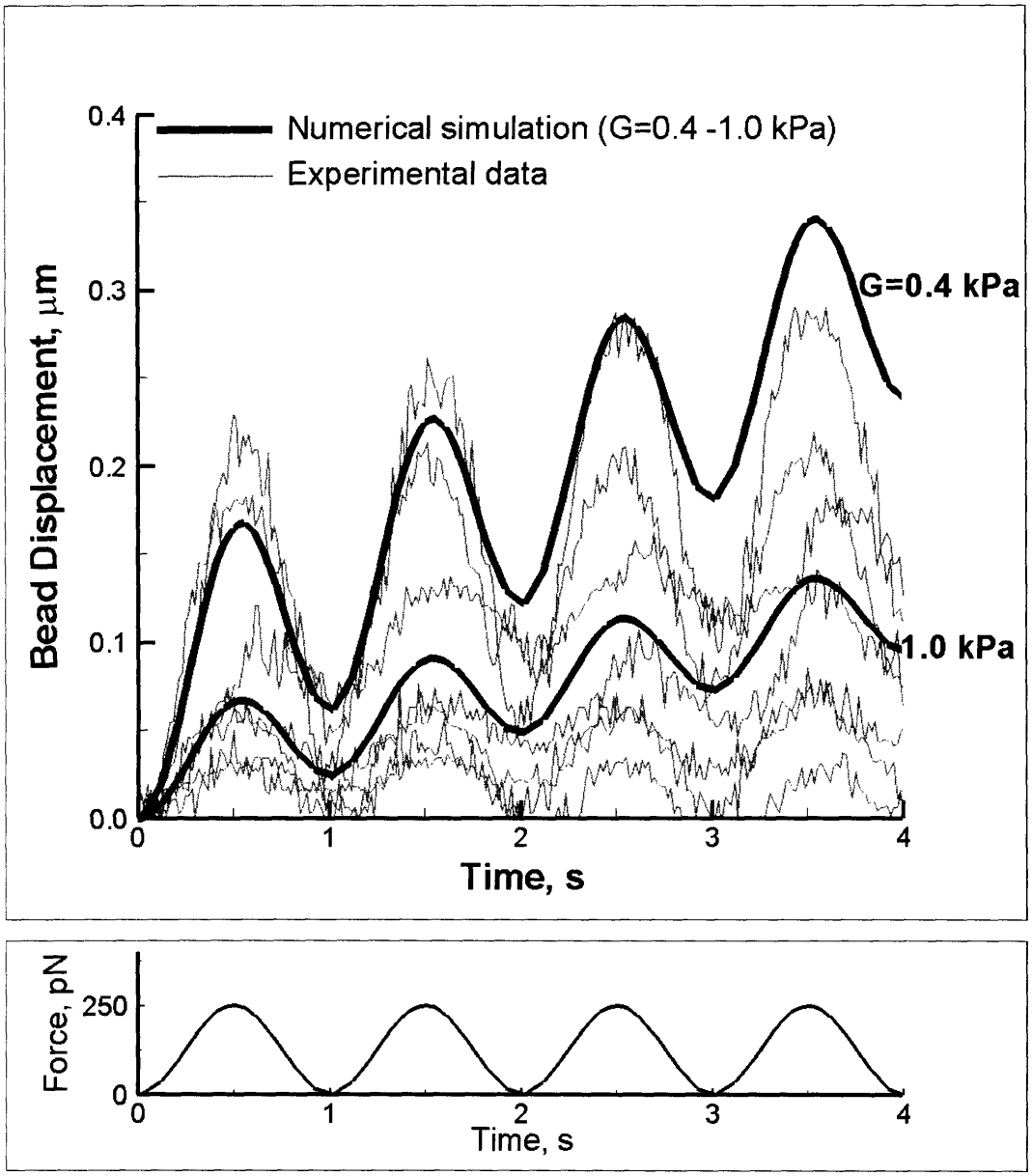


Figure 4.6. Bead center displacement (*upper graph*) as a result of force applied to the bead (*lower graph*) versus time. Four periods of the numerical simulations (black curve) are represented along with the first four periods of sample data from a single experiment featuring 7 beads each attached to a different NIH 3T3 fibroblast (7 gray curves) (see *Cell experiments*).

4.5 References

1. Gullingsrud, J., Kosztin, D. & Schulten, K. (2001) *Biophys. J.* **80,** 2074-2081.
2. Sawada, Y. & Sheetz, M. P. (2002) *J. Cell Biol.* **156,** 609-615.
3. Zhu, C., Bao, G. & Wang, N. (2000) *Annu Rev Biomed Eng.* **2.**
4. Satcher, R. L., Jr. & Dewey, C. F., Jr. (1996) *Biophys J* **71,** 109-18.
5. Ingber, D. E. (1993) *J Cell Sci* **104 (Pt 3),** 613-27.
6. Ingber, D. E. (1997) *Annu Rev Physiol* **59,** 575-99.
7. Stamenovic, D. & Coughlin, M. F. (1999) *J Theor Biol* **201,** 63-74.
8. Bausch, A. R., Ziemann, F., Boulbitch, A. A., Jacobson, K. & Sackmann, E. (1998) *Biophys J* **75,** 2038-49.
9. Karcher, H., Lammerding, J., Huang, H., Lee, R. T., Kamm, R. D. & Kaazempur-Mofrad, M. R. (2003) *Biophys J* **85,** 3336-49.
10. Huang, H., Dong, C. Y., Kwon, H.-S., Sutin, J. D., Kamm, R. D. & So, P. T. C. (2002) *Biophys. J.* **82,** 2211-2223.
11. Bausch, A. R., Moller, W. & Sackmann, E. (1999) *Biophys J* **76,** 573-9.
12. Caille, N., Thoumine, O., Tardy, Y. & Meister, J. J. (2002) *J Biomech* **35,** 177-87.
13. Charras, G. T. & Horton, M. A. (2002) *Biophys J* **82,** 2970-81.
14. Hochmuth, R. M. (2000) *J Biomech* **33,** 15-22.
15. Fabry, B., Maksym, G., Hubmayr, R., Butler, J. & Fredberg, J. (1999) *J Magnetism Magnetic Mat* **194,** 120-125.
16. Mahaffy, R. E., Park, S., Gerde, E., Kas, J. & Shih, C. K. (2004) *Biophys J* **86,** 1777-93.
17. Landau, L. D. & Lifschitz, E. M. (1988) *Mechanics by L.D. Landau and E.M. Lifshitz ; translated from the Russian by J.B. Sykes and J.S. Bell.* (Pergamon Press, New York).
18. Boulbitch, A. A. (1999) *Physical Review E.* **59,** 3402-3407.
19. Laurent, V., Henon, S., Planus, E., Fodil, R., Balland, M., Isabey, D. & Gallet, F. (2002) *Journal of Biomechanical Engineering* **124,** 408-421.
20. Heidemann, S. R., Kaech, S., Buxbaum, R. E. & Matus, A. (1999) *J Cell Biol* **145,** 109-22.
21. Lammerding, J., Schulze, P. C., Takahashi, T., Kozlov, S., Sullivan, T., Kamm, R. D., Stewart, C. L. & Lee, R. T. (2004) *J Clin Invest* **113,** 370-8.
22. Fabry, B., Maksym, G. N., Butler, J. P., Glogauer, M., Navajas, D. & Fredberg, J. J. (2001) *Phys Rev Lett* **87,** 148102.

5 Quantitative measurements of passive mechanical properties in normal and hypertrophic adult mouse cardiac myocytes*

5.1 Introduction

Hypertrophic cardiomyopathy is a genetic disease of the myocardium that is characterized by left ventricular hypertrophy in the absence of increased external loads. Mutations in at least seven genes, all encoding cytoskeletal proteins, have been identified as causes of familial hypertrophic cardiomyopathy (1). Affected cardiomyocytes are significantly enlarged compared to wild type cells and often display severe structural disarray. Functionally, these myocytes have impaired contractility and/or relaxation and altered calcium dynamics. Changes in contractility can be caused by altered sarcomeric function, by modifications in passive cytoskeletal stiffness, or by structural changes in the sarcomere organization. One central question in the study of cardiac hypertrophy is whether mutations leading to cardiomyopathy cause only specific biomechanical alterations associated with the protein's known function or if they also affect other biomechanical properties of the cell through compensation or potential common pathophysiological pathways at some time point in the development of hypertrophy. In order to distinguish between these possibilities, it is necessary to independently quantify changes in sarcomeric force generation, passive cell resistance, cytoskeletal structure, and the coupling between the force generating cytoskeleton and the extracellular environment.

Currently, two methods are widely used to quantify contractility in single cardiomyocytes. In the first method, isolated myocytes are plated on transparent cover slips and are paced using a field stimulator. Changes in cell length are recorded using

---

* Sections of this chapter have been published in the *Engineering in Medicine and Biology Society Magazine*, 2003, Vol. 22, pages 124-127, and presented at the Joint Biomedical Engineering Society / Engineering in Medicine and Biology Society 2002.

high speed video edge detection to determine cell shortening and relaxation velocities as well as fractional shortening. Alternatively, single myocytes can be suspended between a force transducers and an actuator attached to each end of the cell, allowing precise measurements of force generated by the cell. Both methods can be combined with optical systems to simultaneously measure sarcomere length. Force clamp systems have the advantage that sarcomere resting length can be adjusted to physiological levels and that the force generated by the cardiomyocytes can be quantified. In the edge detection method, it is not possible to directly measure the generated force or to apply controlled stresses or strains to the cardiomyocytes, as the only resistance to cell contraction is the stiff substrate the cells are attached to. Only the force clamp method allows measurements of passive mechanical properties by applying forces to the cell and measuring induced myocyte lengthening. However, the measurements only reflect the total force integrated over the entire cell length and cross-section and do not account for local variations. Furthermore, measurements are limited to cytoskeletal stiffness in the longitudinal (i.e. sarcomere) direction, as transverse force application is not possible.

We here present a novel approach to quantify passive mechanical properties in longitudinal and transverse directions in adult cardiomyocytes under physiological conditions. Magnetic bead microrheology is used to force laminin-coated beads bound to the cell surface in transverse or longitudinal direction and cytoskeletal stiffness is inferred by measuring induced bead and cytoskeletal displacement (2). Experimental validation was performed by quantifying transverse isotropic mechanical behavior in cardiac myocytes. Subsequently, we studied the effect of two related cardiomyopathy-causing mutations on cellular mechanical properties.

Desmin, a muscle-specific member of the intermediate filament family, is encoded by a single gene and expressed in striated and smooth muscle cells. In mature striated myocytes, desmin filaments are found at the Z-disc where they interconnect sarcomeres with each other and link the contractile apparatus to the sarcolemmal cytoskeleton, mitochondria and the nucleus (3). A putative 7-amino acid deletion (D7-des) has been linked to desmin-related myopathy, a group of human myopathies

characterized by abnormal intrasarcoplasmic desmin accumulation and often affecting only cardiac and skeletal muscle (4-6). Transgenic mice expressing the D7-des deletion have aberrant intrasarcoplasmic desmin aggregation, perturbed myofibril alignment, defective myocyte mechanical function, and cardiac hypertrophy (7). In contrast, transgenic mice overexpressing wild-type desmin appear normal, indicating that overabundance of desmin can not be responsible for the observed phenotype. Furthermore, contractile and relaxation function of isolated D7-des cardiomyocytes are significantly impaired compared to wild type myocytes (7).

The small heat shock protein αB-crystallin (CryAB) is the most abundant heat shock protein in heart and skeletal muscle and binds to both desmin and cytoplasmic actin (8-10). In response to cellular stress, it transits from the cytosol to the cytoskeleton where it is thought to participate as a chaperone in intermediate filament formation and maintenance (11, 12). CryAB is upregulated in familial hypertrophic cardiomyopathy and overexpression of CryAB has been demonstrated to protect cardiomyocytes from ischemia and reperfusion injury (13, 14). An autosomal dominant missense mutation (R120G) in the CryAB gene has been linked to familial desmin-related cardiomyopathy (15). Transgenic mice overexpressing the R120G-CryAB mutation develop a phenotype similar to that observed for desmin-related cardiomyopathies. Mutant mice have aberrant desmin and CryAB aggregation in cardiac myocytes, impaired myofibril alignment, and develop severe cardiomyopathy (16). Concentric hypertrophy is apparent at 3 months and increases over time with ventricular chamber dilation. R120G-CryAB mutants die between 24-32 weeks of age. In contrast, overexpression of wild-type CryAB has no appears benign, with no increase in mortality or induction of desmin-related cardiomyopathy (16).

Here, we measure passive mechanical properties in the transverse and longitudinal myocyte direction in single adult mouse cardiac myocytes isolated from R120G-CryAB and Des-D7 mutants and wild type litter mates to explore cellular and molecular mechanisms that lead to desmin-related cardiac hypertrophy. We demonstrate that the R120G-CryAB mutation leads to significant changes in cytoskeletal mechanical

properties in the transverse force directions, possibly caused by sarcomeric disarray or by desmin and CryAB aggregates within the cytoskeleton.

5.2 Materials and methods

Desmin wild type and D7-des mutant mice. Transgenic mice expressing either murine wild type desmin or a 7–amino acid deletion (R173 through E179) desmin mutation (D7-des) linked to desmin-related myopathy were obtained from Jeffrey Robbins (Childrens Hospital Medical Center, Division of Molecular Cardiovascular Biology, 3333 Burnett Avenue, Cincinnati, OH 45229). Transgenic mice were made and kept in the FVB/n background (7). Heterozygous wild type desmin or D7-des male mice were crossed with wild-type female animals to produce wild type and heterozygous offspring. Mice were genotyped by collecting buccal cell DNA and amplifying desmin DNA using polymerase chain reaction (PCR) with the following primers: Desmin-forward: 5' cag ctt cag gaa cag cag gtc c -3'; desmin-reverse: 5' cat caa tct cgc agg tgt agg act g -3'. Annealing temperature was 65°C and the final $MgCl_2$ concentration was 2.5 mM. Experiments were performed on cells isolated from 64 ± 1 week old mice.

αB-Crystallin wild type and R120G-CryAB mutant mice. Transgenic mice overexpressing either murine wild-type CryAB or the R120G mutation in cardiomyocytes were obtained from Jeffrey Robbins (see address above). Transgenic mice were made and kept in the FVB/n background (16). Heterozygous wild type CryAB or R120G-CryAB male mice were crossed with wild-type female animals to produce wild-type and heterozygous offspring. Mice were genotyped by extracting DNA from buccal cells and amplifying CryAB DNA using PCR with the following primers. CryAB-forward: 5' ctg gcg ttc ttc gtg ctt gcc gtg -3'; CryAB-reverse: 5' gag tct gac ctc ttc tca aca gcc -3'. Annealing temperature was 65°C at a final $MgCl_2$ concentration of 1.5 mM. Experiments were performed on cells isolated from 20 ± 2 week old mice.

Magnetic bead experiments. Local mechanical properties were evaluated by applying small forces to laminin-coated paramagnetic beads (Dynabeads M-450, Dynal) bound to

the cell surface and measuring induced bead displacement and cytoskeletal deformation (2). Adult mouse cardiac myocytes were isolated using a standard perfusion protocol (17), plated on laminin (L2020, Sigma, 10 μg/ml in PBS) coated culture dishes (Corning, Corning, NY) and incubated at 37°C in myocyte plating medium for one hour to achieve firm cell attachment of viable cardiac myocytes (see Appendix D for details). Subsequently, cells were washed once with myocyte culture medium and media was replaced with a suspension of laminin-coated paramagnetic microbeads in myocyte culture medium (final bead concentration of ~6 × 10^5 beads per dish). Sufficient bead attachment was achieved after incubation for 1 h at 37° C. A sinusoidal force (amplitude 0.6 nN, offset 0.6 nN, 1 Hz) was applied to microbeads on the cell surface using a computer controlled magnetic trap. Induced bead and cytoskeletal displacements were imaged through an inverted light microscope (Olympus America, Melville, NY), recorded using digital cameras (ES310T and CoolSNAP HQ, Roper Scientific, San Diego, CA), and stored on a personal computer for further processing. Experiments were performed on a temperature controlled stage and limited to 10-20 minutes per dish to minimize changes in buffer conditions. Custom-written MATLAB software (MathWorks, Natick, MA) was used to analyze bead and cytoskeletal displacements. Magnetic bead position was computed from digital videos based on a center-of-mass algorithm with better than 10 nm resolution. Cytoskeletal and single sarcomere displacements were analyzed as described below. Displacement-time curves were subsequently analyzed using Fast-Fourier-Transformation (FFT) and autocorrelation algorithms to compute displacement amplitude, frequency and phase lag with respect to the forcing function.

Cytoskeletal displacement measurements. Cytoskeletal displacements were computed using a normalized cross-correlation algorithm. The cross-correlation algorithm was confined to 1-pixel resolution for large displacement maps, corresponding to ~300 nm resolution. For quantitative measurements of cytoskeletal displacements, cross-correlation resolution was improved to ~0.1 pixel (~30 nm) resolution by fitting a 2^{nd} order polynomial to the 9 points surrounding the peak of the normalized cross-correlation function. Mean cytoskeletal displacement was evaluated based on the displacements of 6 regions distributed evenly around a circle with a radius of 6.6 μm centered on the

magnetic bead. Each region consisted of an 11×11 pixel square kernel, corresponding to a ~3.3×3.3 μm^2 area on the cell surface. Regions not located on the cell or showing cross-correlation artifacts (e.g. zero image intensity standard deviation, low normalized cross-correlation maximum, edge effects, or non-continuous jumps in position) were excluded. The displacement-time curves obtained from the valid regions generally showed excellent agreement with each other (see Fig.. 5.6) and were averaged to obtain a single cytoskeletal displacement curve.

Single sarcomere displacement measurements. Single sarcomere positions within short (~10-15 μm) myofibril segments were detected by analyzing the line intensity profile along a user defined axis. In order to improve sarcomere detection, corresponding line intensities for parallel line segments within 1 μm of the user defined segment were also computed and averaged to obtain a one-dimensional intensity profile along the selected myofibril segment. In these intensity profiles, the sarcomere striation pattern appears as an oscillating intensity profile, with each peak representing a single sarcomere and Z-bands marked by regions of low image intensity. Sarcomere centroid positions along the line segment were computed using a one-dimensional center of mass algorithm and tracked from frame to frame, providing single sarcomere measurements with sub-pixel resolution.

Magnetic trap calibration. Magnetic trap calibration was performed as described previously (18). In brief, 4.5-μm diameter paramagnetic beads (Dynal, Dynabeads M-450, Oslo, Norway) were suspended in a highly viscous fluid (Dimethylpolysiloxane, Sigma, DMPS – 12M) and tracked while being attracted to the magnetic trap operated at a range of electrical currents. The applied force for a given current was computed as a function of distance from the magnetic trap based on Stokes equation, $F=3\pi \nu D u$, where ν denotes the viscosity of the fluid, D the bead diameter and u the relative velocity between fluid and bead.

Cell size measurements. Cell lengths and widths were measured in the same adult mouse cardiac myocytes used for magnetic bead experiments. Images of cells were taken

at 10× magnification and analyzed using custom-written software. Reported values are based on ~100-200 cells per animal.

Fluorescence labeling. For live cell organelle fluorescence staining, mitochondria were labeled with Mitotracker Green or Rhodamine 123 (both Molecular Probes, Eugene, OR). Nuclei were labeled with Hoechst 33342 nuclear stain (Molecular Probes). Fluorescence images of live cells were acquired using a digital camera (Roper CoolSNAP HQ) or a custom-build two photon imaging system.

Statistics. All results are expresses as mean ± standard error. Comparisons between groups were performed using a student t-test for two groups or ANOVA for more than two groups. A two-tail p-value of less than 0.05 was considered significant.

5.3 Results

To validate our methods, we examined the anisotropic mechanical behavior of wild type adult mouse cardiac myocytes. Transverse isotropic material behavior in striated muscle cells is based on the highly oriented sarcomere/myofibril structure but is difficult to quantify using existing methods. Analyzing bead displacement under two perpendicular force directions, we found that both bead displacement amplitude and mean displacement were significantly smaller in the sarcomere (longitudinal) direction compared to the transverse direction (Fig. 5.1), indicating that the cytoskeleton is significantly stiffer in the sarcomere direction. Furthermore, the phase lag between applied force and bead displacement was significantly increased in the transverse direction, suggesting a larger viscous component in the mechanical response.

Since bead displacement is a combination of cytoskeletal displacement and the effects of mechanical coupling between the bead and the cytoskeleton, we evaluated the extent of cytoskeletal displacement using a 2-D cross-correlation algorithm. Figure 5.2 shows two representative displacement maps at the onset of force application. For transverse force application, large areas of the CSK showed induced displacement,

whereas CSK displacement was not detectable for force application parallel to the sarcomere direction.

The cross-correlation algorithm used above was constrained to 1-pixel resolution, corresponding to 300 nm resolution at 30× magnification. In contrast, single sarcomere position and length can be monitored with better than 20 nm resolution by using a modified center-of-mass algorithm for single sarcomeres and monitoring their relative position over time. Using this high-resolution algorithm, we found that cytoskeletal deformations decayed rapidly along the sarcomere direction. Force induced sarcomere displacement was only detectable in close proximity to the magnetic bead, while sarcomeres more than a few micrometers away from the magnetic bead did not show induced displacements (Fig. 5.3).

Following the validation experiments, we applied our experimental methods to adult mouse cardiac myocytes isolated from D7-des and R120G-CryAB mutant mice and wild type littermates. Mutant myocytes had considerable cellular hypertrophy, evidenced by significantly increased cell width compared to wild type littermates (Fig. 5.4). Myocyte length was not significantly affected in either mutant compared to wild type littermates (data not shown).

Magnetic bead experiments revealed altered cytoskeletal mechanics in the R120G-CryAB mutants compared to wild type littermates. In the transverse force direction, R120G-CryAB myocytes exhibited substantially smaller bead displacement amplitudes compared to wild type littermates (Fig. 5.5 A), although the difference did not quite reach statistical significance ($P = 0.0699$). However, when grouping together both wild-type groups from CryAB and Desmin mice to achieve a larger sample number, the difference became statistically significant. In contrast, D7-des mutants had no significant difference in bead displacement amplitude compared to wild type littermates or compared to the combined group of wild type mice. Force application in sarcomere direction did not yield any statistically significant differences in bead displacement amplitudes for any group (Fig. 5.5. B). As demonstrated in the previous sections, wild type cardiac myocytes

showed anisotropic material behavior with significantly smaller bead displacement in sarcomere direction compared to the transverse direction. However, this difference was less distinct for the R120G-CryAB mutant, indicating a potential loss in uni-directional myofibril alignment (compare Figs. 5.5 A, B).

Interestingly, R120G-CryAB mutants also showed significant differences in viscoelastic behavior for transverse force application (Fig. 5.5 C). The phase lag between applied force and observed bead displacement was significantly reduced in mutant myocytes compared to cells derived from wild type litter mates, indicating a reduction in viscous cell behavior. D7-des mutants did not show any differences in viscoelastic bead behavior, and viscous bead behavior was similar for all cells under force application in sarcomere direction. Differences in residual bead displacement were not significant for any group, regardless of the direction of force application.

Since bead displacement is a combination of bulk cytoskeletal displacement and bead movement relative to cytoskeleton through bead rolling and local membrane strain, we measured induced cytoskeletal displacement in close proximity (~6 μm) to the bead (Fig. 5.6). For transverse force application, cytoskeletal displacement amplitude was significantly reduced in R120G-CryAB mutants compared to wild type litter mates, indicating significantly increased cytoskeletal stiffness in transverse cell direction (Fig. 5.7 A). In addition, phase lag in the transverse direction was also significantly reduced in R120G-CryAB mutants, closely matching the results from the bead displacement analysis. The difference in cytoskeletal displacement amplitude and phase lag in the transverse direction cannot simply be explained by the increase in cell width of the R120G-CryAB cells, as D7-des mutants have similarly increased cell widths but do not exhibit reduced cytoskeletal displacements compared to wild type littermates.

Force application in the sarcomere direction did not reveal any significant differences in cytoskeletal displacements. For all cell types, cytoskeletal displacement was significantly larger for transverse force application compared to force application in

sarcomere direction (Fig. 5.7 C), confirming the anisotropic behavior observed in the magnetic bead displacements.

5.4 Discussion

Previous experiments have shown that accumulation of microtubules in cardiac myocytes can impede sarcomere motion and contribute to increased ventricular stiffness in hypertrophy and heart failure (6, 19, 20). Here, we demonstrate that ultrastructural changes in cytoskeletal organization observed in R120G-CryAB mutant cardiac myocytes are associated with impaired cytoskeletal mechanics in the transverse direction. Adult mouse cardiac myocytes isolated from R120G-CryAB mutants have sarcomeric disarray and disruption of the normal desmin network, characterized by the absence of the normal sarcomeric striation pattern for desmin filaments, the presence of desmin and αB-crystallin aggregates, perturbation in the alignment of adjacent myofibrils at the Z-band, and increased and non-uniform Z-band thickness (16). In our experiments, laminin-coated magnetic beads bound to surface of isolated cardiac myocytes displaced significantly less under transverse force application, and induced cytoskeletal displacement was significantly reduced compared to wild type littermates, indicating significantly increased transverse cytoskeletal stiffness. Wild-type myocytes have uniformly oriented myofibrils aligned at the Z-band through desmin filaments, resulting in anisotropic cell mechanics with a significantly increased stiffness in the longitudinal direction compared to the transverse direction (see Figs. 5.1 and 5.5 A,B). In contrast, R120G mutant cells show less obvious anisotropic mechanics and the transverse stiffness approaches the level of the longitudinal stiffness. The increase in transverse stiffness cannot be attributed to changes in cell geometry, as the R120G mutants are comparable in size to the D7-des mutant cells, but the D7-des mutants do not show the same impaired cytoskeletal mechanics. Instead, we speculate that altered sarcomeric alignment and cytoplasmic desmin and αB-crystallin aggregates function as lateral reinforcement or cross-linker in the R120G mutant cells, comparable to the function of diagonal struts in stabilizing a scaffold, resulting in increased transverse stiffness.

In addition to the increased transverse cytoskeletal stiffness, we found that R120G mutant cells had decreased cytoskeletal viscosity, characterized by a significant reduction in the phase lag between force application and induced displacement. Phase lag values for magnetic bead displacement and cytoskeletal displacement were nearly identical for transverse force application, indicating that bulk cell movement with a relatively large time constant dominates the viscous bead response. The reason for the loss of transverse cytoskeletal viscosity is not clear, but most likely reflects the same ultrastructural changes described above, i.e. increased lateral reinforcement and cross-linking of myofibrils. However, since the phase lag is determined by the ratio of elastic and viscous mechanical contribution, the increase in phase lag could also be dominated by the increase in elastic stiffness.

The consequences of impaired transverse passive mechanical properties on physiological myocyte function are not yet clear, but they could result in impaired contraction/relaxation dynamics. Wang et al. (16) previously reported significantly compromised cardiac function in the R120G-CryAB mutant mice. In vivo hemodynamic measurements in 6 months old mice revealed that R120G-CryAB mutants were unable to maintain normal contractility and had a significant, load-independent increase in relaxation time. Contractile function measured at 3 months in isolated work-performing hearts also revealed significantly impaired relaxation in the mutant mice compared to wild-type or WT-CryAB control cohorts. While ventricular viscoelastic behavior is dominated by the extracellular matrix (21), increased cytoskeletal stiffness caused by altered sarcomere alignments and protein aggregates could additionally contribute to the impaired contraction and relaxation dynamics. The correlation between the amount of sarcomeric disarray and desmin/CryAB aggregates and the increase in transverse cytoskeletal stiffness when comparing the R120G-CryAB and D7-des mutants suggests that these ultrastructural changes are at least in part responsible for the observed changes in cytoskeletal stiffness and the more severe phenotype of the R120G-CryAB mutation.

Nonetheless, altered ventricle contraction/relaxation and changes in cytoskeletal displacement under force could also be caused by differences in myocyte activation or

altered calcium dynamics, as we performed our experiments at physiological extracellular calcium concentrations and did not actively block actin/myosin interactions to study passive stiffness under physiological conditions. Additional experiments using altered calcium concentrations or actin-myosin inhibitors such as 2,3-Butanedione monoxime (BDM) could help distinguish between the individual contributions of actin-myosin cross-bridges in quiescent myocytes and actin/myosin-independent passive stiffness.

In our experiments, we found that bead displacement amplitudes were significantly larger than the corresponding cytoskeletal displacements, indicating a significant amount of bead movement relative to the cell. This relative bead movement can be caused by highly localized deformations and membrane strain at the force application site (see Chapter 3) or by partial bead detachment and rolling. Localized deformations that rapidly decay away from the bead were confirmed in the single sarcomere observations. Bulk cytoskeletal displacement on the other hand can be caused by shearing the entire cell or large regions under force and is very uniform over large cell areas. Displacements measured in 6 small regions evenly distributed around the magnetic bead at a distance of ~6 μm showed generally very uniform displacement values. This bulk shear movement is independent of bead movement relative to the cell, as it only depends on the applied force, and can be used to measure the cellular shear modulus G. The bulk shear modulus can be estimated based on the deformation and the applied strain by the following relationship

$$G = \frac{\tau}{\gamma}, \qquad (1)$$

where G is the bulk shear modulus, τ is the applied shear stress and γ is the shear deformation. Assuming a uniform force distribution over the entire cell cross-sectional area, the applied shear stress can be approximated by $\tau = F/A$, where F is the applied force and A is the longitudinal cross-sectional area of a myocytes, i.e. length (L) × width (W). For small deformations, the shear deformation can be expressed as $\gamma = \Delta x/h$, where Δx is the displacement at the top cell surface and h is the cell height. Substituting these expressions into equation 1, we obtain the following expression for the apparent bulk shear modulus.

$$G = \frac{F}{A(\Delta x / h)} = \frac{F}{WL(\Delta x / h)} \qquad (2)$$

Using typical values obtained from our experiments and assuming a cell height of ~15 μm, equation 2 yields a bulk shear modulus of ~23 Pa for wild type cells and a bulk shear modulus of ~44 Pa for the R120G-CryAB mutant. These estimates have to be regarded as lower bounds of the actual bulk shear modulus, as they assume a uniform stress distribution over the entire cross-sectional area and ignore the viscous contribution of the cytoskeletal mechanics under dynamic load. The bulk shear modulus in the transverse direction derived from these experiments is almost three orders of magnitude lower than the transverse stiffness values reported from AFM experiments on cardiac and skeletal muscle cells (22-24). However, it is important to notice that the AFM experiments use small indentations (<100 nm) with a very small probe tip (r ~ 10 nm) and therefore probe the transverse stiffness of individual myofibrils, while our experiments compute the bulk shear modulus of the entire cell. As an analogy, one can envision the adult cardiac myocyte as a stack of carefully aligned ropes, where each rope represents a myofibril. The transverse stiffness of each rope is significantly smaller than its longitudinal stiffness (e.g. ~10 kPa vs. ~100 kPa for myofibrils) but still much higher than the bulk shear modulus of the entire stack. Consequently, our experiments reflect how well the myofibrils are connected to each other, as this macro-architecture determines the bulk mechanical properties. Nonetheless, AFM could provide a valuable complement to our experiments to measure transverse mechanical properties, as AFM can be applied to measure localized effects of the protein aggregates on cellular mechanics. Previous studies have demonstrated the general feasibility of AFM measurements on adult cardiac myocytes, although these experiments were performed on myocytes from different species such as rat, cat, or rabbit that are not as sensitive as adult mouse cardiac myocytes (22-24).

The highly organized cytoskeletal structure of adult cardiac myocytes immediately suggests an anisotropic material behavior, but most current techniques fail to provide accurate measurements of anisotropic mechanical properties. Our methods allow independent measurement of the cytoskeletal stiffness in arbitrary directions and thus can

quantify the cytoskeletal properties more accurately. Single sarcomere position measurements allow high resolution measurement of CSK displacement and single sarcomere contractility. Furthermore, single sarcomere length measurements can be used to assess the effect of local external force application on sarcomere contractility. The active sarcomere contraction in field stimulated myocytes against an externally applied force (through the magnetic bead) yields an indirect estimate of the active contractile force. An inherent disadvantage of brightfield microscopy is the obstruction of the view on underlying CSK structures through the magnetic bead, limiting the area under investigation to regions at least a few micrometers away from the magnetic bead. Considering the rapid decay of induced displacements (Fig. 5.3), it is desirable to directly observe sarcomeres underneath the bead. This can be achieved by tracking the displacement of fluorescently labeled mitochondria that are tightly coupled to the CSK (figure 5.8). Additionally, induced deformations of the fluorescently labeled nucleus can be imaged, enabling the concomitant analysis of nuclear mechanics in wild-type and mutant cells (25, 26). Unfortunately, larger exposure times necessary for fluorescence microscopy limit the temporal resolution. Furthermore, the application of magnetic traps can easily be expanded to probe the connection between the extracellular matrix and the cytoskeleton. Recently, we demonstrated that cells with a mutation in the integrin associated tetraspanin CD151 show increased bead detachment rates compared to wild type cells (27). These techniques can now be applied to variety of cardiac myocytes derived from genetically engineered mice to assess the effect of mutations on cellular mechanics and cardiac remodeling.

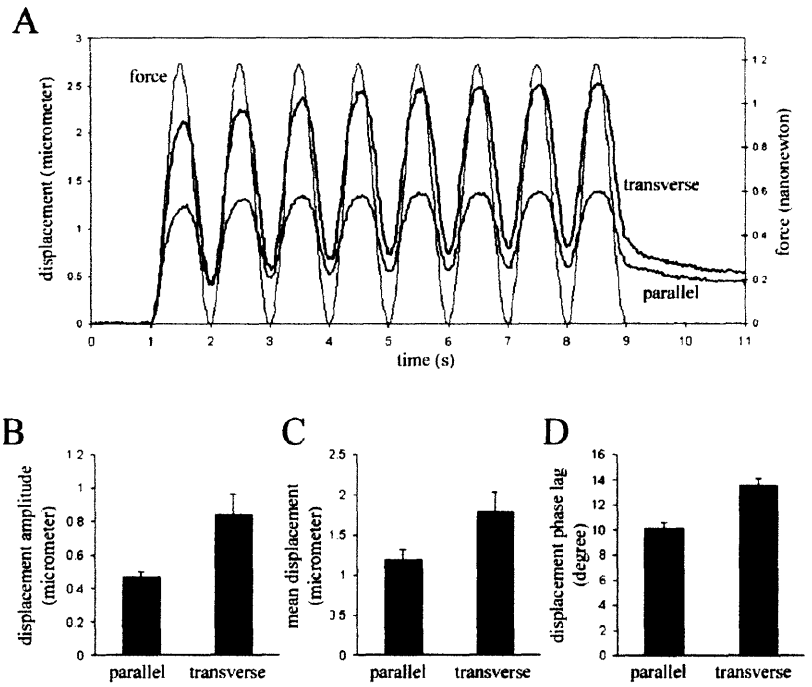


Figure 5.1. Anisotropic material behavior in adult mouse cardiac myocytes. (A) Time course of magnetic bead displacement for parallel (red) and transverse (blue) force application. Displacement amplitude, mean displacement, and phase lag can be inferred from these plots. (B) Bead displacement amplitude was significantly larger under transverse forcing (0.469 ± 0.031 μm vs. 0.841 ± 0.121 μm, $P < 0.01$). (C) Bead mean displacement was significantly increased in transverse direction (1.190 ± 0.124 μm vs. 1.782 ± 0.246 μm, $P < 0.05$). (D) Bead displacement phase lag was significantly increased (10.07 ± 0.55 ° vs. 13.46 ± 0.61 °, $P < 0.001$).

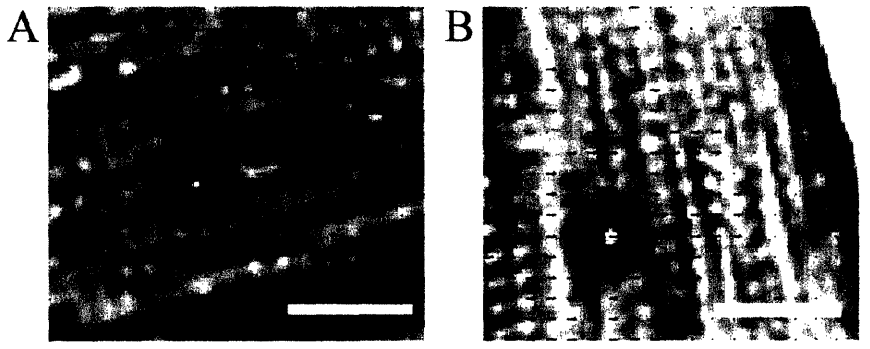


Figure 5.2. Displacement map at onset of sinusoidal force application. (A) Force application parallel to sarcomere direction. All vectors in the image center are associated with bead movement. (B) Force in transverse direction. Direction of force is to the right. Vector scale 2×, scale bar 10 μm.

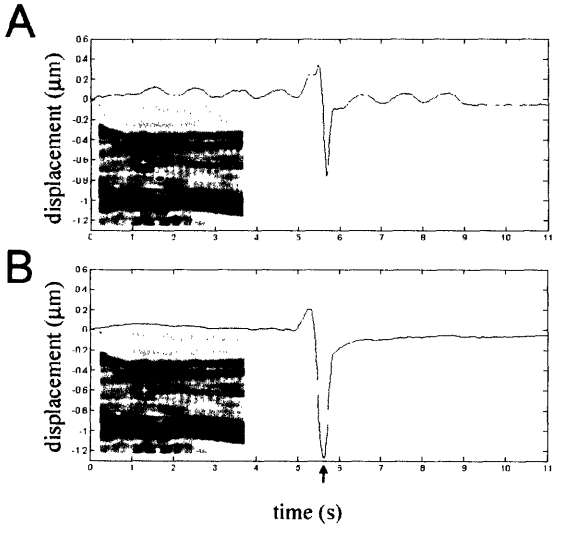


Figure 5.3. Single sarcomere displacement analysis. (A) Induced displacement in sarcomeres close to the magnetic bead. Onset of sinusoidal force at 1 sec, end at 9 sec. Arrow denotes spontaneous myocyte contraction. (B) No detectable induced displacement in sarcomeres away from magnetic bead, except during myocyte contraction. (insets) Yellow cross denotes magnetic bead center, red circle marks position of tracked sarcomere.

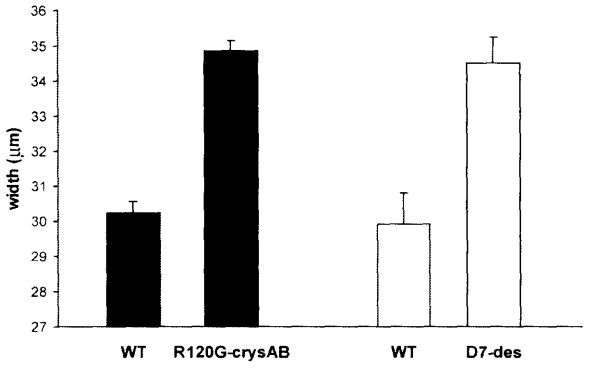


Figure 5.4. Cell width measurements for mutant and wild-type cardiac myocytes. Cell width for R120G-CryAB and D7-des mutants was significantly increased compared to wild type littermates (34.9 ± 0.3 μm vs. 30.25 ± 0.3 μm for R120G-CryAB and WT littermates respectively, $P < 0.0001$; 34.5 ± 0.7 μm vs. 29.9 ± 0.9 μm for D7-des and WT littermates respectively, $P < 0.0001$). Cell length did not vary significantly between mutants and wild type littermates

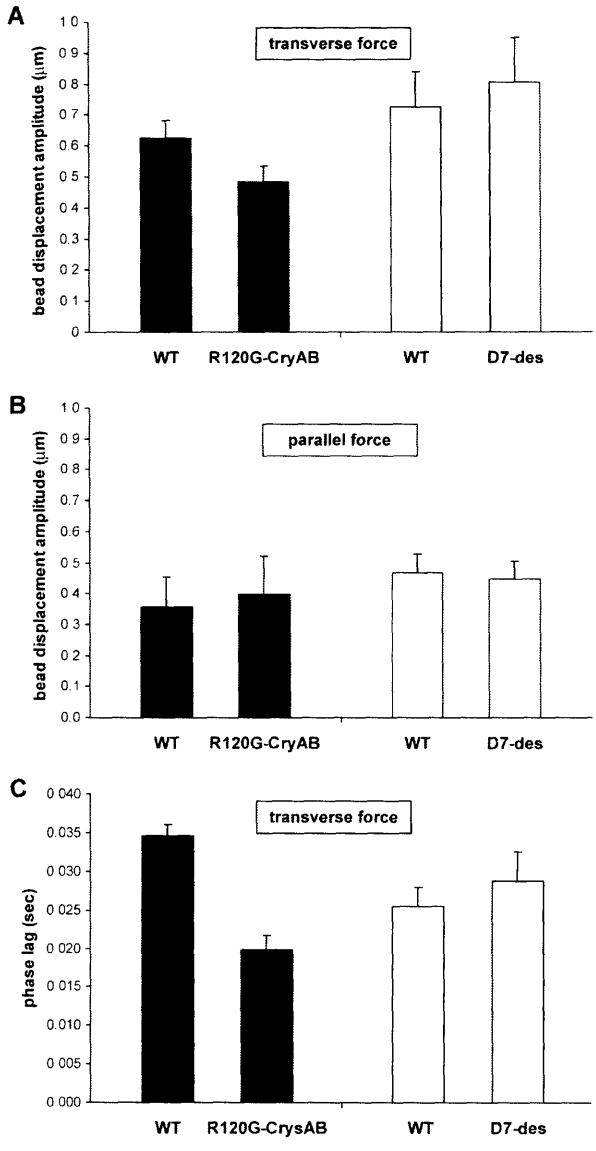


Figure 5.5. Magnetic bead displacement amplitude and phase lag are impaired in R120G-CryAB but not in D7-des mutants. (A) Magnetic bead displacement amplitude is reduced in R120G-CryAB mutants for force application in transverse force direction (black bars, 0.483 ± 0.050 μm vs. 0.623 ± 0.056 μm for R120G-CryAB and WT littermates respectively, $P = 0.0699$). Combining wild type groups from both animal colonies yields bead displacement amplitudes of 0.659 ± 0.055 μm vs. 0.483 ± 0.050 μm and 0.806 ± 0.143 μm for wild-type, R120G-CryAB and D7-des mice respectively; P = 0.0211 for R120G-CryAB vs. wild-type). Magnetic bead displacement amplitudes for force

application in sarcomere direction (white bars) is not significantly different. (B) Phase lag between forcing function and magnetic bead displacements for force application in transverse direction. Phase lag is significantly reduced in R120G-CryAB mutants (0.020 ± 0.002 s vs. 0.035 ± 0.002 s for R120G-CryAB and WT littermates respectively, $P < 0.001$).

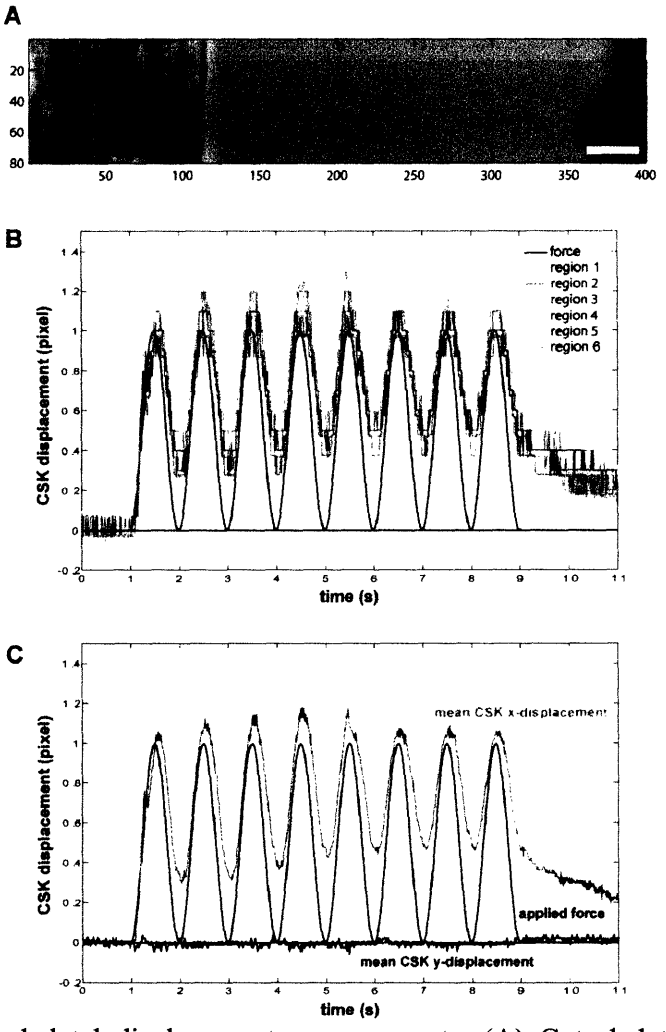


Figure 5.6. Cytoskeletal displacement measurements. (A) Cytoskeletal displacements were measured in 6 square regions of ~3×3 μm^2 evenly distributed around the magnetic bead center at a distance of ~6 μm. The magnetic trap tip is visible at the far right edge of the image, applying a sinusoidal forcing function in the horizontal (x-coordinate) direction. Scale bar 10 μm. (B) Cytoskeletal displacement in force direction for the 6 regions depicted above. The graph demonstrates the uniform cytoskeletal displacement (colored curves) around the bead. The forcing function is plotted as reference (black curve); peak force corresponds to 1.2 nN. (C) Cytoskeletal displacement in x- (green) and y-direction (red) averaged over the 6 regions. The forcing function is plotted as reference (black). Mean cytoskeletal displacement in x-direction follows the forcing function with a small phase lag. Virtually no displacement is detectable in the y-direction.

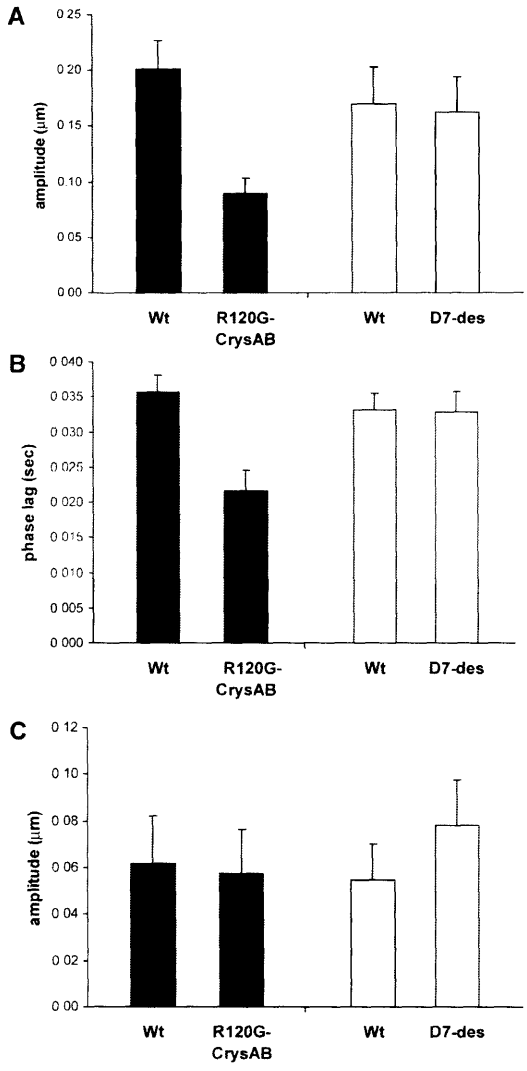


Figure 5.7. R120G-CryAB mutants have altered cytoskeletal mechanics. (A) Cytoskeletal displacement amplitude for transverse force application. Cytoskeletal displacement amplitude is significantly reduced in R120G-CryAB mutants (0.090 ± 0.014 μm vs. 0.201 ± 0.026 μm for R120G-CryAB and WT littermates respectively, $P < 0.001$). (B) Phase lag of cytoskeletal displacement relative to forcing function for transverse force application. Phase lag is significantly reduces in R120G-CryAB mutants (0.022 ± 0.003 s vs. 0.036 ± 0.002 s for R120G-CryAB and WT littermates respectively, $P < 0.001$). (C) Cytoskeletal displacement amplitude for force application in sarcomere direction.

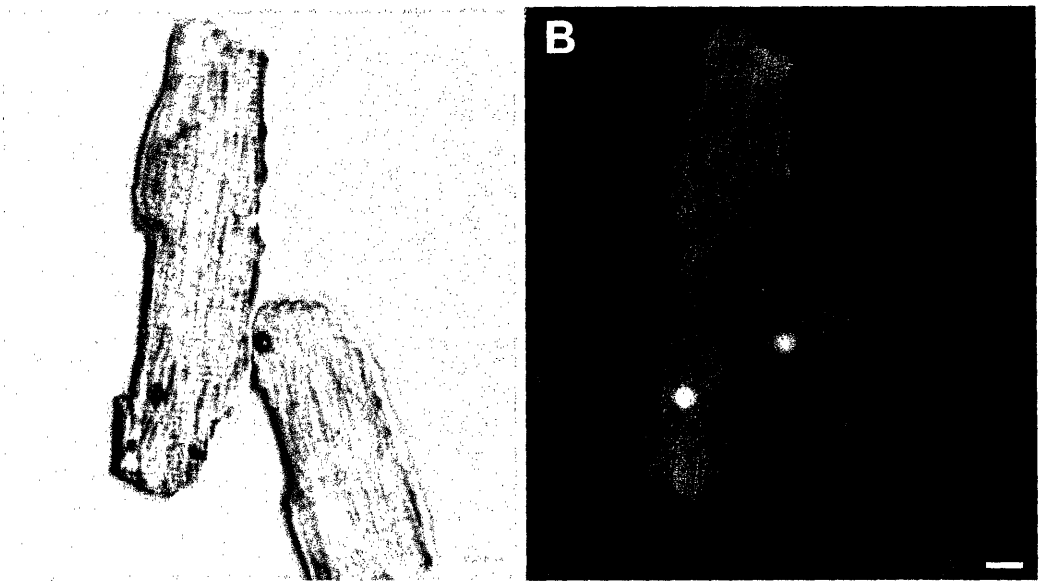


Figure 5.8. Adult mouse cardiac myocyte imaged using bright field and fluorescence microscopy. (A) Cardiomyocytes imaged under bright field. Two magnetic beads can be seen on the cells. (B) Fluorescence image of the same cells. Mitochondria were stained in green, nuclei in blue and magnetic beads in red. Association of mitochondria to desmin filaments and the Z-disks results in visible striation pattern. Scale bar 10 μm.

5.5 References

1. Towbin, J. A. (1998) *Curr Opin Cell Biol* **10,** 131-9.
2. Bausch, A. R., Ziemann, F., Boulbitch, A. A., Jacobson, K. & Sackmann, E. (1998) *Biophys J* **75,** 2038-49.
3. Milner, D. J., Taffet, G. E., Wang, X., Pham, T., Tamura, T., Hartley, C., Gerdes, A. M. & Capetanaki, Y. (1999) *J Mol Cell Cardiol* **31,** 2063-76.
4. Munoz-Marmol, A. M., Strasser, G., Isamat, M., Coulombe, P. A., Yang, Y., Roca, X., Vela, E., Mate, J. L., Coll, J., Fernandez-Figueras, M. T., Navas-Palacios, J. J., Ariza, A. & Fuchs, E. (1998) *Proc Natl Acad Sci U S A* **95,** 11312-7.
5. Goldfarb, L. G., Park, K. Y., Cervenakova, L., Gorokhova, S., Lee, H. S., Vasconcelos, O., Nagle, J. W., Semino-Mora, C., Sivakumar, K. & Dalakas, M. C. (1998) *Nat Genet* **19,** 402-3.
6. Heling, A., Zimmermann, R., Kostin, S., Maeno, Y., Hein, S., Devaux, B., Bauer, E., Klovekorn, W. P., Schlepper, M., Schaper, W. & Schaper, J. (2000) *Circ Res* **86,** 846-53.
7. Wang, X., Osinska, H., Dorn, G. W., 2nd, Nieman, M., Lorenz, J. N., Gerdes, A. M., Witt, S., Kimball, T., Gulick, J. & Robbins, J. (2001) *Circulation* **103,** 2402-7.
8. Iwaki, T., Kume-Iwaki, A., Liem, R. K. & Goldman, J. E. (1989) *Cell* **57,** 71-8.
9. Bennardini, F., Wrzosek, A. & Chiesi, M. (1992) *Circ Res* **71,** 288-94.
10. Longoni, S., Lattonen, S., Bullock, G. & Chiesi, M. (1990) *Mol Cell Biochem* **97,** 121-8.
11. Nicholl, I. D. & Quinlan, R. A. (1994) *Embo J* **13,** 945-53.
12. Barbato, R., Menabo, R., Dainese, P., Carafoli, E., Schiaffino, S. & Di Lisa, F. (1996) *Circ Res* **78,** 821-8.
13. Xiao, X. & Benjamin, I. J. (1999) *Am J Hum Genet* **64,** 685-90.
14. Arbustini, E., Morbini, P., Grasso, M., Fasani, R., Verga, L., Bellini, O., Dal Bello, B., Campana, C., Piccolo, G., Febo, O., Opasich, C., Gavazzi, A. & Ferrans, V. J. (1998) *J Am Coll Cardiol* **31,** 645-53.
15. Vicart, P., Caron, A., Guicheney, P., Li, Z., Prevost, M. C., Faure, A., Chateau, D., Chapon, F., Tome, F., Dupret, J. M., Paulin, D. & Fardeau, M. (1998) *Nat Genet* **20,** 92-5.
16. Wang, X., Osinska, H., Klevitsky, R., Gerdes, A. M., Nieman, M., Lorenz, J., Hewett, T. & Robbins, J. (2001) *Circ Res* **89,** 84-91.
17. Sambrano, G. R., Fraser, I., Han, H., Ni, Y., O'Connell, T., Yan, Z. & Stull, J. T. (2002) *Nature* **420,** 712-4.
18. Lammerding, J., Kazarov, A. R., Huang, H., Lee, R. T. & Hemler, M. E. (2003) *Proc Natl Acad Sci U S A* **100,** 7616-21.
19. Tagawa, H., Wang, N., Narishige, T., Ingber, D. E., Zile, M. R. & Cooper, G. t. (1997) *Circ Res* **80,** 281-9.
20. Ishibashi, Y., Takahashi, M., Isomatsu, Y., Qiao, F., Iijima, Y., Shiraishi, H., Simsic, J. M., Baicu, C. F., Robbins, J., Zile, M. R. & Cooper, G. t. (2003) *Am J Physiol Heart Circ Physiol* **285,** H1270-85.
21. Stroud, J. D., Baicu, C. F., Barnes, M. A., Spinale, F. G. & Zile, M. R. (2002) *Am J Physiol Heart Circ Physiol* **282,** H2324-35.
22. Mathur, A. B., Collinsworth, A. M., Reichert, W. M., Kraus, W. E. & Truskey, G. A. (2001) *J Biomech* **34,** 1545-53.
23. Nyland, L. R. & Maughan, D. W. (2000) *Biophys J* **78,** 1490-7.
24. Lieber, S. C., Aubry, N., Pain, J., Diaz, G., Kim, S. J. & Vatner, S. F. (2004) *Am J Physiol Heart Circ Physiol.*
25. Maniotis, A. J., Chen, C. S. & Ingber, D. E. (1997) *Proc Natl Acad Sci U S A* **94,** 849-54.
26. Burke, B., Mounkes, L. C. & Stewart, C. L. (2001) *Traffic* **2,** 675-83.
27. Lammerding, J., Kazarov, A. R., Huang, H., Lee, R. T. & Hemler, M. E. (in press) *Proc Natl Acad Sci U S A.*

6 Lamin A/C deficiency causes defective nuclear mechanics and mechanotransduction*

6.1 Introduction

Lamins are structural components of the nuclear lamina, a protein network underlying the inner nuclear membrane that determines nuclear shape and size (1). In addition, lamins play an important role in organizing nuclear pore complexes (2) and recruiting other proteins such as emerin to the nuclear envelope (3, 4). Two types of lamins are found in mammalian cells: A-type lamins (lamin A, C, AΔ10 and C2) are encoded by a single gene (*Lmna*) and are developmentally regulated and expressed in differentiated cells. B-type lamins (B1 and B2/B3) are encoded by two distinct genes and are constitutively expressed in all cells (1, 5-10). Mutations in the gene encoding A-type lamins and their binding partners have been associated with a variety of human diseases, including Emery-Dreifuss muscular dystrophy (EDMD), dilated cardiomyopathy (DCM), Dunnigan-type familial partial lipodystrophy (FPLD) and Hutchinson-Gilford progeria syndrome(11-17).

The molecular mechanisms underlying the varied phenotypes are unknown, and two alternative hypotheses have been proposed to explain the tissue specific effects observed in laminopathies. The "structural hypothesis" suggests that lamin mutations lead to increased nuclear fragility and eventual nuclear disruption in the mechanically stressed tissue, while the "gene regulation hypothesis" proposes a tissue-specific role of lamins in DNA transcription. *Lmna*$^{-/-}$ mice are indistinguishable from their littermates at birth but develop severe muscle wasting and contractures similar to EDMD by 3-4 weeks and die by eight weeks (4). Cells derived from *Lmna*$^{-/-}$ mice have misshaped nuclei and obvious ultrastructural damage (4, 18). Distorted nuclear shape has also been demonstrated in fibroblasts from lipodystrophic patients with heterozygous R482Q/W

* This chapter has been published in the *Journal of Clinical Investigation* 2004, Vol. 113(3), pp. 370-378 and has been presented at the American Heart Association Scientific Sessions 2003 and the Novartis Foundation Symposium on nuclear envelope in development and disease 2004.

mutations in the lamin A/C gene and in cells from C. elegans with reduced lamin levels (1, 19). Here we show that nuclear mechanics in cells from *Lmna*$^{-/-}$ mice are defective, with *Lmna*$^{-/-}$ nuclei displaying increased deformations and fragility under strain. In addition, we demonstrate that transcriptional activation in response to mechanical stimuli is attenuated in *Lmna*$^{-/-}$ cells, impairing viability of mechanically strained cells. These data suggest that the structural and gene regulation hypotheses of the laminopathies are, in fact, closely related, and different mutations may cause specific phenotypes by differentially affecting these processes.

6.2 Materials and methods

Cells. *Lmna*$^{+/+}$ and *Lmna*$^{-/-}$ mouse embryo fibroblasts were maintained in DMEM (Invitrogen, Carlsbad, CA) containing 10% FCS (HyClone, Logan, UT) and penicillin/streptomycin (Invitrogen).

Nuclear strain experiments. Cells were plated (900 cells/ cm^2) on fibronectin coated silicon membranes in DMEM supplemented with 10% FCS followed by serum starvation for 48 h in DMEM containing ITS supplement (Sigma, St. Louis, MO). Preceding the strain experiments, cells were incubated with Hoechst 33342 nuclear stain (1 μg/ml, Molecular Probes, Eugene, OR) in DMEM + ITS for 20 min. Membranes were placed on a custom-made strain device mounted on an Olympus IX-70 microscope and biaxial strain was applied in a stepwise fashion. Membrane and nuclear strains were computed on brightfield and fluorescence images using a custom written image analysis algorithm. Normalized nuclear strain was defined as the ratio of nuclear strain to membrane strain to compensate for small variations in applied membrane strain (range 17.4 -19.8 %).

Magnetic bead microrheology. Cells were plated on 35 mm polystyrene dishes (Corning, Corning, NY). The following day, cells were incubated with fibronectin coated paramagnetic beads (Dynal Biotech, Lake Success, NY) for 30 min. To minimize nuclear effects, only beads attached more than 5 μm from the nucleus were selected for analysis. A sinusoidal force (amplitude 0.6 nN, 1 Hz, offset 0.6 nN) was applied through a magnetic trap and bead displacement was monitored using a digital camera (Roper

Scientific, San Diego, CA). Displacement amplitudes were computed using custom-written MATLAB (Mathworks) algorithms. In separate experiments, smaller (2μm) fibronectin coated polystyrene beads (Bangs Laboratories, Fishers, IN) were incubated together with the magnetic beads for 1 hour to adhere to the cell surface. Cells containing single magnetic beads and several polystyrene beads were subjected to a brief force pulse (2.5 nN, 3 sec). Maximal induced magnetic and polystyrene bead displacements were computed and expressed in cylindrical coordinates (r, ϕ)with the magnetic bead at the origin using custom-written MATLAB algorithms. The induced strain field can be described by an analytical cell mechanics model proposed by Bausch et al. (20), expressing the radial component u_r of the induced bead displacement as a function of the applied force F, cell stiffness μ^*, the characteristic cut of radius κ^{-1} , the distance from the magnetic bead center r and the angle ϕ in respect to the force direction:

$$u_r(r) = \frac{F}{2\pi\mu^*}\cos(\phi)\left\{\frac{3(1-\sigma)}{4}\mathrm{K}_0(\kappa_1 r) - \frac{\mathrm{K}_1(\kappa r)}{\kappa r} + \sqrt{\frac{1-\sigma}{2}}\frac{\mathrm{K}_1(\kappa_1 r)}{\kappa r}\right\} \qquad (1)$$

where K_0 and K_1 are modified Bessel functions of the second kind (order zero and one respectively) and using $\kappa_1 = [(1-\sigma)/2]^{1/2}\kappa$. The parameters μ* and κ were obtained by fitting equation 1 to the bead displacement data using the GraphPad Prism 4.0 (GraphPad Software, San Diego, CA) robust curve fit function and assuming σ = 0.5 for incompressible media and a magnetic bead contact radius of 2 μm.

The magnetic trap calibration was performed as described previously (21). In brief, magnetic beads suspended in viscous solution were monitored while being attracted to the magnetic trap operated at various currents. The applied force as a function of current and distance from the magnetic trap was then computed based on Stoke's Law.

Microinjection. Cells were plated on fibronectin coated glass dishes (WillCo Wells, Amsterdam, Netherlands) or silicon dishes and incubated overnight. Microinjections were performed using an Eppendorf microinjector (Eppendorf, Hamburg, Germany) with Eppendorf Femtotips. In each dish, 20-50 cells were injected with TexasRed-labeled 70kDa Dextran (Molecular Probes, dissolved at 10 mg/ml in PBS (Invitrogen)) into the

cytoplasm (500 hPa, 0.6 sec) or into the nucleus (10, 100, 500, and 1500 hPa, 0.6 sec). Following the microinjection, cells were washed in HBSS (Invitrogen) and intracellular localization of Dextran-TexasRed was recorded under a fluorescent microscope. Selected silicon membranes were subjected to constant (ca. 32%, 30 min) or cyclic bi-axial strain (10%, 1 Hz, 23 hours) and localization of TexasRed labeled dextran in strained and control cells was analyzed on a fluorescence microscope (Olympus, Melville, NY).

Strain experiments. Strain stimulation was carried out as previously described(22). In brief, cells were plated on fibronectin-coated silicon membranes (2,500-3,300 cells /cm^2). After 72 h serum starvation, cells were subjected to bi-axial cyclic strain (4% or 10%, 1 Hz). For chemical stimulation, cells were incubated with IL-1β (25 ng/ ml, R&D Systems, Minneapolis, MN) or PMA (200 ng/ml, Sigma) in DMEM + ITS.

DsRed-Peroxiredoxin-2 localization. Cells were plated on fibronectin glass dishes or and fibronectin coated silicon membranes. Following overnight incubation, cells were transfected with a CMV-promoter driven DsRed-peroxiredoxin-2 fusion construct (ClonTech, Palo Alto, CA) using FuGene6 (Roche, Basel, Switzerland) and incubated for 24 hours. Selected silicon membranes were subjected to constant (ca. 19%, 60 min) or cyclic bi-axial strain (10%, 1 Hz, 3 hours) and localization of DsRed labeled peroxiredoxin-2 in strained and control cells was analyzed on a fluorescence microscope (Olympus).

Flow cytometry. For cell viability assays, propidium iodide (PI, Sigma, 2 μg/ml) was added to the dishes after 24 h strain application. Cells were collected and analyzed using a Cytomics FC 500 flow cytometer (Beckman Coulter, Fullerton, CA), counting 10-30,000 events in each group. Thresholds for PI incorporation were determined based on negative (no PI staining) and positive (cells permeabilized by 50% ethanol) controls. Apoptotic and necrotic cell fractions were measured in similar experiments using the Vybrant Apoptosis Assay Kit #3 (Molecular Probes).

Northern and Western analyses. Expression of *iex-1* and *egr-1* mRNA was assessed by Northern analysis as described previously (23). Protein expression was analyzed by Western analysis of nuclear and cytoplasmic cell fractions using antibodies against NF-κB p65 (antibody does not recognize p50 or p105), IκBα (both Santa Cruz Biotechnology, Santa Cruz, CA), and actin (Sigma). Additional immunoblotting was performed on whole cell lysates using specific antibodies against total ERK1/2 (Santa Cruz Biotechnology) and phospho-p44/p42 MAP Kinase (ERK1/2; Cell Signaling Technology, Beverly, MA). After incubation with horseradish peroxidase (HRP)-conjugated secondary antibody (Bio-Rad Laboratories, Hercules, CA.), specific bands were visualized by enhanced chemiluminescence (Perkin Elmer, Boston, MA).

Luciferase experiments. Cells were transfected with plasmids for NF-κB-controlled luciferase expression and SV40-regulated β-galactosidase (Promega, Madison, WI) using FuGene6 (Roche). Following transfection, cells were serum starved in DMEM + ITS medium for 48 h, followed by overnight stimulation with PMA (200 ng/ml) or IL-1β (25 ng/ml). Luciferase assays were quantified in a Victor2 Multilabel Counter (Perkin Elmer). Results were normalized for β-galactosidase activity and expressed as per cent wildtype control.

Immunohistochemistry. Cells were plated on untreated or fibronectin coated chambers slides and serum-starved for 24-72 h followed by stimulation with IL-1β. Cells were fixed in 4% paraformaldehyde or methanol, washed in PBS and permeabilized by 0.1% Triton X 100. After blocking, cells were incubated overnight with primary antibodies (rabbit anti-NF-κB (p65, Santa Cruz) at 4° C or Alexa Fluor 568 phalloidin (Molecular Probes A-12380), 1 h at 25° C), followed by 1 h incubation with secondary FITC or TRITC conjugated antibodies.

Cellular protein fractions and electromobility shift assay (EMSA). Nuclear extracts were prepared as described previously(24) with the following modifications. Cells were washed with ice-cold PBS and lysed in buffer A (0.1% Triton X-100, 10 mM EDTA, 10 mM EGTA, 10 mM KCl, 10 mM HEPES, 1 mM DTT, 0.5 mM PMSF and protease

inhibitor cocktail (Sigma)). After centrifugation (1,200 x g, 10 min), the supernatant was stored as cytoplasmic cell fraction, while the nuclear pellet was washed once in PBS, resuspended in buffer C (1 mM EDTA, 1 mM EGTA, 0.4 M NaCl, 20 mM HEPES, 5 mM $MgCl_2$, 25% glycerol, 1 mM DTT, 0.5 mM PMSF, and protease inhibitor cocktail), and incubated at 4 °C for 10 min. The nuclear extract was centrifuged for 10 min at 4 °C at 12,000 x g, and supernatant was used for western analysis and electrophoretic mobility shift assay (EMSA). NF-κB specific oligonucleotides (Promega) were end-labeled using T4 polynucleotide kinase and [γ–^{32}P]ATP (DuPont/NEN, Boston, MA). Nuclear extracts were preincubated for 10 min in binding buffer followed by 20 min incubation at room temperature with labeled oligonucleotide. Samples were separated on a 4% native polyacrylamide gel. For competition studies, 50X excess unlabeled oligonucleotide was used, and for the supershift assay, the nuclear extracts were incubated with 2 μg of anti-p50 or anti-p65 antibody (Santa Cruz).

Statistical analysis. All experiments were performed at least three independent times. Data are expressed as mean ± SEM. Statistical analysis was performed using the PRISM 4.0 and INSTAT software (GraphPad, San Diego, CA). The data were analyzed by unpaired t-test (allowing different SD), one-way ANOVA or the Mann-Whitney test in case of non-parametric distribution. A two-tailed P-value of <0.05 was considered significant.

6.3 Results

Nuclear mechanics. To explore the role of lamin A/C in nuclear mechanics, mouse embryo fibroblasts derived from *Lmna*$^{-/-}$ and *Lmna*$^{+/+}$ mice were cultured on transparent membranes and subjected to stepwise increasing biaxial strain (step 1: 10.1 ± 0.18 %, step 2: 18.2 ± 0.07%). The cytoskeleton, attached to the membrane through integrin receptors, is exposed to the same bi-axial strain as the membrane, while the stiffer nucleus exhibits only small deformations in wild type cells (25). The induced nuclear deformations were calculated by tracking distinct features in the fluorescently labeled chromatin (Figure 6.1, a-b) and normalized to membrane strain to compensate for the

small variation in the applied membrane strain. For each cell type, nuclear deformation increased approximately linearly with applied membrane strain (Figure 6.1c), but *Lmna*$^{-/-}$ nuclei showed significantly larger deformations compared to wild type cells (Figure 6.1, c-d). Fitting a linear regression to the nuclear deformation data revealed a significantly larger slope for the *Lmna*$^{-/-}$ nuclei, and the maximal normalized nuclear deformation was significantly larger for *Lmna*$^{-/-}$ cells, indicating impaired nuclear mechanics in lamin A/C deficient nuclei.

Cytoskeletal mechanics. To evaluate the possibility that the observed increase in nuclear deformation was caused by more direct force transmission to the nucleus, we examined cytoskeletal organization by staining actin stress fibers with phalloidin. No apparent differences in cytoskeletal architecture were found between wild type (Figure 6.2a) and *Lmna*$^{-/-}$ cells (Figure 6.2b). Magnetic bead microrheology (20) was utilized for a quantitative evaluation of cytoskeletal stiffness. Small (4.5 μm), fibronectin-coated paramagnetic beads were attached to the cell and the bead displacement in response to an applied magnetic force was used as an indicator of cytoskeletal stiffness (Figure 6.2, c). The induced bead displacement amplitude was significantly increased in *Lmna*$^{-/-}$ fibroblasts (Figure 6.2d), indicating decreased cytoskeletal stiffness. Since the magnetic bead displacement depends not only on the cytoskeletal stiffness, but also the binding characteristics of the cell to the bead , we applied a second microrheology method that measures the induced displacement in polystyrene beads located on the cell surface close to the magnetic bead (Figures 6.2e-f). Lamin A/C deficient cells exhibited significantly larger induced polystyrene bead displacement that decreased quickly with increasing distance from the magnetic bead, whereas bead displacements in wild type cells were smaller and hardly exceeded the detection limit of less than 0.1 μm (Figures 6.2g-h). Based on an analytical model proposed by Bausch et al. (20), the induced polystyrene bead displacement is a function of the applied force, the position relative to the magnetic bead, and two cellular mechanics parameters μ^* and κ that describe the cell stiffness and intracellular strain dissipation respectively. By separately fitting the theoretical displacement field to the observed bead data for *Lmna*$^{-/-}$ and wild type fibroblasts one can

estimate the parameters for cellular stiffness and dissipation (Figures 6.2g-h), confirming that *Lmna*$^{-/-}$ cells exhibit significantly decreased cytoskeletal stiffness.

Therefore, the observed increased nuclear deformation in lamin deficient cells is unlikely to be due to altered force transmission to the nucleus. In contrast, the softer cytoskeleton would result in an underestimation of the nuclear stiffness. Based on these findings we conclude that lamin A/C deficient cells have decreased nuclear stiffness and altered nuclear mechanics.

Cellular response to mechanical strain. Strain-induced damage to the more fragile nucleus could provide one explanation for tissue-specific effects of lamin A/C mutations, for example, in mechanically active tissues like myocardium and skeletal muscle. To examine nuclear envelope integrity, we monitored the subcellular location of fluorescently labeled 70 kDa dextran microinjected either into the cytoplasm or nucleus of adherent fibroblasts. Cytoplasmic injection revealed that the high molecular weight dextran was excluded from the nucleus in both wild type and lamin deficient cells (Figure 6.3a-b, e), indicating that nuclear integrity is not significantly impaired in *Lmna*$^{-/-}$ cells under resting conditions. These results were confirmed by transfecting the cells with a DsRed-Peroxiredoxin-2 fusion protein. The protein lacks a nuclear localization sequence and is too large (MW 57 kDa) to passively diffuse into the intact nucleus. Consequently, the fusion protein was excluded from the nucleus of wild type and *Lmna*$^{-/-}$ cells.

However, when dextran was injected directly into the nucleus at low and medium injection pressures (10-500 hPa), nuclear integrity in lamin A/C deficient was at least temporarily compromised, resulting in fluorescently labeled dextran escaping into the cytoplasm (Figure 6.3d). In contrast, dextran was confined to the nucleus in most wild type cells when injected at low pressure (Figure 6.3c), and even at medium pressure significantly more wild type cells maintained their nuclear integrity compared to *Lmna*$^{-/-}$ cells. Using sufficiently high pressure (1,500 hPa), both wild type and lamin deficient nuclei could be ruptured (Figure 6.3e).

To determine if *Lmna*$^{-/-}$ fibroblasts are thus more susceptible to mechanical strain, cells were subjected to cyclic bi-axial strain (10% strain, 1 Hz). After 24 h of strain, *Lmna*$^{-/-}$ fibroblasts had a significantly increased fraction of dead (i.e. propidium iodide-positive) cells compared to unstrained controls (Figure 6.4a). Differences within controls (wild type and *Lmna*$^{-/-}$) and between wild type control and strained cells were not significant. Dual labeling with propidium iodide and a FITC conjugated annexin V antibody revealed that the decrease in viability was due to an increase in both necrotic and apoptotic cell fractions compared to wild type cells (Figure 6.4b).

Interestingly, when cells were injected with the fluorescently labeled 70 kDa dextran into the cytoplasm and subjected to either constant (30% for 30 min) or cyclic (10%, 1Hz, 24 h) bi-axial strain, strain application did not significantly increase the number of dextran positive nuclei in either wild type or *Lmna*$^{-/-}$ fibroblasts (data not shown), suggesting that the extreme event of nuclear rupture under mechanical strain occurs only in a small fraction of cells that cannot be detected based on a small number (n = 20-30) of single cell observations.

The increased fraction of apoptotic cells in mechanically strained *Lmna*$^{-/-}$ fibroblasts indicates that necrosis through nuclear rupture can only in part explain the increased sensitivity to mechanical stimulation. Therefore, we investigated the cellular response to mechanical stimulation in more detail. Interestingly, expression of the mechano-sensitive genes *egr-1* and *iex-1* in response to mechanical stimulation was impaired in *Lmna*$^{-/-}$ cells (Figure 6.4c), whereas expression of the mechanically unresponsive genes *thioredoxin-1* and *glyceraldehyde 3-phosphate dehydrogenase* (*GAPDH*) was unaltered (data not shown), indicating that transcription is not impaired in a nonspecific manner. To test if the observed changes in *Lmna*$^{-/-}$ cells were specific to mechanical stimulation or if they represented a more general deficiency in signal transduction, we measured the expression levels of *iex-1* and *egr-1* in response to stimulation with PMA or the cytokine IL-1β. Cytokine stimulation led to an attenuated response in *iex-1* but not *egr-1* expression in *Lmna*$^{-/-}$ cells, while PMA-responsiveness remained intact for both genes in *Lmna*$^{-/-}$ cells (Figure 6.4d).

NF-κB signaling. Because *iex-1* is an NF-κB dependent survival gene (23), and because NF-κB can be biomechanically activated (26), we examined if biomechanical signaling through NF-κB is disturbed in *Lmna*$^{-/-}$ cells. The mitogen activated protein kinase (MAPK) ERK 1/2 is an important regulator for mechanically induced gene expression and has been linked to NF-κB activation (27). Analysis of MAPK phosphorylation after chemical (PMA) or mechanical stimulation revealed no differences between *Lmna*$^{-/-}$ and wild type cells (data not shown), indicating that the observed changes are caused by alterations in signal transduction other than impaired cytoplasmic MAPK activation.

In resting cells, NF-κB is sequestered in the cytoplasm by the inhibitor IκB. Upon stimulation, IκB is ubiquinated and degraded, allowing NF-κB to translocate into the nucleus and activate target genes. Figure 6.5a shows that cytokine induced cytoplasmic degradation of IκBα and translocation of the NF-κB subunit p65/RelA into the nucleus was not impaired in *Lmna*$^{-/-}$ cells. This finding was confirmed in immunofluorescence directed against p65/RelA (data not shown). Total levels of NF-κB subunits p50, p65, and p105 in whole cell extracts from Lmna$^{-/-}$ cells were indistinguishable from wild type cells (data not shown). Interestingly, binding of nuclear NF-κB to its transcription factor binding site was increased in the lamin A/C deficient cells compared to wild type cells (Figure 6.5b). Surprisingly, despite the increased levels of transcription factor binding, IL-1β-induced activity of NF-κB dependent luciferase was significantly impaired in *Lmna*$^{-/-}$ fibroblasts (Figure 6.5c). These results indicate that the deficient response of *Lmna*$^{-/-}$ cells to mechanical or cytokine stimulation is based on a role of lamin A/C in transcriptional activation following transcription factor binding.

6.4 Discussion

Our results establish the importance of lamin A/C for nuclear stability and highlight its role in transcriptional regulation in response to mechanical or chemical stimulation. Measurements on nuclear mechanics were obtained by using a novel technique to measure nuclear deformation with bi-axial strain applied to the cells. This

method yields quantitative measurements of nuclear stiffness compared to cytoskeletal stiffness in living cells without having to isolate the nuclei. Our results for nuclear deformations in strained *Lmna*$^{-/-}$ cells are in excellent agreement with measurements obtained by Caille et al. (25) in endothelial cells under uni-axial strain and on isolated endothelial cell nuclei (28). Increased nuclear fragility in *Lmna*$^{-/-}$ cells was further confirmed by nuclear microinjection experiments, which demonstrated increased nuclear rupture at low and moderate pressures compared to wild type cells. The markedly increased nuclear deformability and fragility in *Lmna*$^{-/-}$ cells compared to wild type cells has important implications for the cellular response to mechanical strain. Impaired nuclear stability can lead to rupture of the nucleus resulting in direct cell death. Evidence of fragmented nuclei has been reported in skeletal muscle fibers from emerin deficient EDMD patients and in fibroblasts from FPLD patients following heat shock treatment (19, 29). This direct effect of impaired nuclear stability is also consistent with the impaired cell viability after cyclic strain observed in our experiments.

In addition to the direct effect of nuclear rupture, altered nuclear mechanics can affect cells through impaired nuclear mechanotransduction. Recent studies report desmin intermediate filament mediated changes in chromatin in response to mechanical strain and hypothesized that stretch-induced changes in chromatin can lead to activation of hypertrophy-associated genes (30). Mechanical connections between integrins, cytoskeletal filaments and the nucleus have also been demonstrated by micromanipulation with microbeads and micropipettes in endothelial cells (31). Our nuclear strain experiments demonstrate that external strain application results in increased nuclear strain, indicating mechanical coupling between the extracellular matrix and the nucleus mediated through focal adhesion sites and the cytoskeleton. Mutations in nuclear envelope proteins such as lamin or emerin could interrupt some of these connections and impair nuclear mechanotransduction pathways. In addition, the impaired nuclear and cytoskeletal mechanics observed in *Lmna*$^{-/-}$ cells lead to significantly increased nuclear strain, which could further result in altered nuclear mechanosensing.

The observation of decreased cytoskeletal stiffness in *Lmna*$^{-/-}$ cells raises additional questions. Alterations in cytoskeletal stiffness could arise as a compensatory mechanism to protect a more fragile nucleus, but could also play a pivotal role in the pathophysiology of the disease. Altered cytoskeletal mechanics not only affect the transmitted force to the nucleus under applied strain, but can also play an important role in cell shape, migration, and other critical functions with direct consequences on the affected tissue.

However, increased nuclear fragility can only partially explain the phenotypes observed in several laminopathies. Even though nuclear mechanics are clearly impaired in the vast majority of *Lmna*$^{-/-}$ cells, only a very small fraction of cells (ca. 3-5%) exhibited nuclear rupture under strain in our experiments, and only a small number of ruptured nuclei are found in affected muscle tissue in patients suffering from EDMD (29). In addition, fibroblasts obtained from lipodystrophic patients with the lamin R482Q/W mutation also exhibit defective nuclear mechanics, even though these patients lack a muscular phenotype (19). Therefore, most cells appear to be functional despite distorted nuclear shape and altered nuclear mechanics, and additional events might be necessary to trigger nuclear failure. Therefore, defective nuclear mechanics could play a more important role in muscle tissue that is subjected to large mechanical strains and stress compared to adipose tissue for example, with cumulative cellular damage, both through direct nuclear rupture and impaired mechanotransduction signaling, eventually leading to the observed muscular dystrophy in several laminopathies.

In addition to their role as nuclear envelope proteins, lamins form stable structures within the nucleoplasm as shown by fluorescence recovery after photobleaching (FRAP)(32).The impaired response of *iex-1* and *egr-1* expression to mechanical and cytokine stimulation as well as the attenuated response of NF-κB regulated luciferase activity in *Lmna*$^{-/-}$ fibroblasts despite increased transcription factor binding indicate an important role of lamin A/C in transcriptional activation. Interactions between lamin and nuclear transcription factors have been previously demonstrated *in vivo* and *in vitro* (33, 34) and lamin A/C speckles can mediate spatial organization of splicing factor

compartments and RNA polymerase II transcription (35). Disrupting the normal organization of nuclear lamins by expression of a dominant negative lamin mutant has been shown to inhibit RNA polymerase II-dependent transcription in mammalian cells and active embryonic nuclei from Xenopus laevis(36). The impaired transcriptional activation observed in our experiments indicates a yet unknown role of lamin A/C in the assembly of enhancesomes or as a scaffolding protein for transcription factors and co-activators.

NF-κB is a mechanical stress-responsive transcription factor that can function as an anti-apoptotic signal. Impaired transcriptional activation can therefore lead to increased apoptosis in mechanically strained tissue. Furthermore, direct evidence for the role of lamin in apoptosis has previously been demonstrated in cultured cells that expressed an uncleavable mutant form of lamin (37). In these cells, chromatin failed to condense and DNA cleavage was delayed despite the activation of caspases. Kumar et al. demonstrated in *ex vivo* experiments that mechanical stress activated NF-κB in skeletal muscle fibers and that this response is altered in mdx mice, a model for Duchenne Muscular Dystrophy (DMD), stressing the importance of NF-κB signaling in muscle tissue that is affected most often in laminopathies (38).

Interestingly, several cells such as lymphoblasts, basal skin cells and early embryonic cells do not express lamin A/C. We speculate that nuclear mechanics and mechanotransduction in these cells would be normal, since these cells are known to be capable of normal function. In these cells, the role of lamin A/C might be taken over by a different protein or complex of proteins. However, it is important to note that our experiments used only fibroblasts and future studies of cells from different tissues are essential.

While all experiments were performed in fibroblasts that completely lack lamin A and lamin C, most human diseases arise from heterozygous lamin mutations, in which case mutant lamin often appear as stable as wild type lamin and is expressed at similar levels (39). In many cases, especially in phenotypes involving striated muscle, the

mutation may lead to a structurally impaired form of lamin A/C that could act as a dominant negative and lead to cellular mechanical deficiencies as observed in the lamin A/C null cells. Other mutations, such as those resulting in lipodystrophy, could affect the binding of lamin to other proteins or chromatin with fewer effects on the structural role of lamin itself, resulting in a partially functional protein that might only affect specific signaling pathways. Furthermore, combinations of mechanical and transcriptional regulation defects could result in complex phenotypes affecting several tissue types in diseases such as progeria. By providing independent tests for measuring structural and gene-regulatory functions of lamin A/C, our experiments could help distinguish the effect of individual mutations on the function of lamin.

Tissue specific effects observed in several laminopathies may thus arise from two mechanisms. (i) The impaired nuclear stability renders mechanically strained tissue more susceptible to cellular damage and (ii) abnormal transcriptional activation impairs adaptive and protective pathways. Individual mutations in the lamin A/C gene could potentially selectively interfere with any of these functions, explaining the diversity of observed phenotypes.

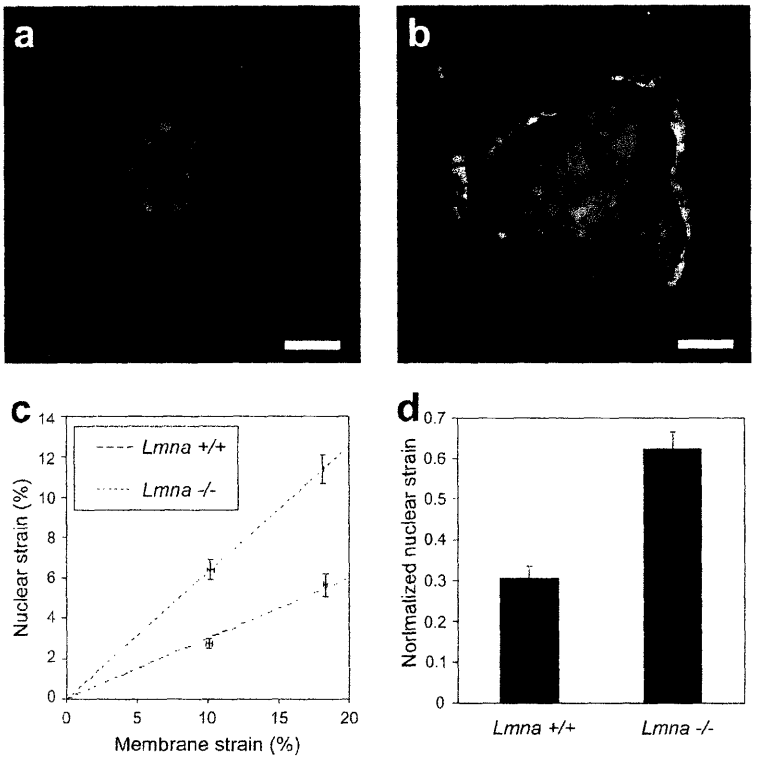


Figure 6.1. Nuclear mechanics is impaired in lamin A/C-deficient cells. (a) Nucleus of wild type fibroblast before (red) and at 22% strain (yellow), scale bar 10 μm. (b) *Lmna*$^{-/-}$ nucleus before (red) and at 19% strain (yellow), scale bar 10 μm. (c) Nuclear deformation as a function of applied membrane strain. Dashed lines represent linear regression of the data for each cell typed forced through the origin (*Lmna*$^{+/+}$: y = 0.299 x, *Lmna*$^{-/-}$: y = 0.626 x). (d) Maximal normalized nuclear strain was significantly increased in *Lmna*$^{-/-}$ fibroblasts (0.306 ± 0.029 vs. 0.626 ± 0.039; $p<0.0001$, n = 21).

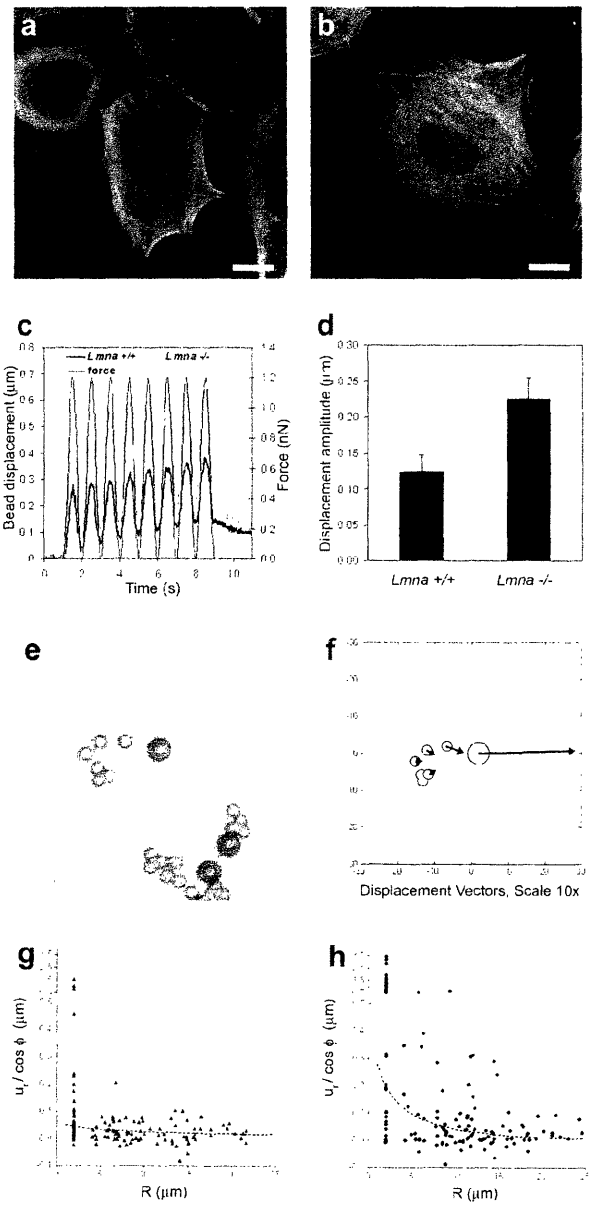


Figure 6.2. Cytoskeletal stiffness is reduced in lamin A/C deficient cells. (a) Phalloidin staining for actin stress fibers in wild type (*Lmna*$^{+/+}$) fibroblasts, scale bar 20 μm. (b) Phalloidin staining for actin stress fibers in *Lmna*$^{-/-}$ cells, scale bar 20 μm. (c) Magnetic bead microrheology. Representative examples of magnetic bead displacement in response to applied sinuisoidal force (black) for wild type (green) and *Lmna*$^{-/-}$ fibroblasts. (d) Bead displacement amplitude in response to applied magnetic forces was significantly increased in *Lmna*$^{-/-}$ fibroblasts, indicating reduced cytoskeletal stiffness in lamin A/C-

deficient cells (0.124 ± 0.024 μm vs. 0.226 ± 0.029 μm, $p < 0.01$, $n = 60$). (e) Fibroblast with magnetic (diameter 4.5 μm) and polystyrene beads (diameter 2μm) attached to the cell membrane, scale bar 10 μm. (f) Graphic representation of the displacement field after a brief force pulse (2.5 nN, 3 sec). Bead sizes and positions are drawn to scale, while bead deflections are enlarged by a factor of 10. (g,h) Distance dependence of the angle corrected radial bead displacement component $u_r / \cos(\phi)$ as defined in equation 1. The dotted line is an optimal fit to equation 1, yielding estimates for cellular stiffness μ* and dissipation κ for wild type and *Lmna*$^{-/-}$ cells respectively (μ*: 27,537 ± 8,458 pN/μm vs. 2,417 ± 734.7 pN/ μm, $p < 0.01$, $n = 128, 153$; k: 0.020 ± 0.017 μm^{-1} vs. 0.201 ± 0.072 μm^{-1}, $p < 0.05$, $n = 128, 153$).

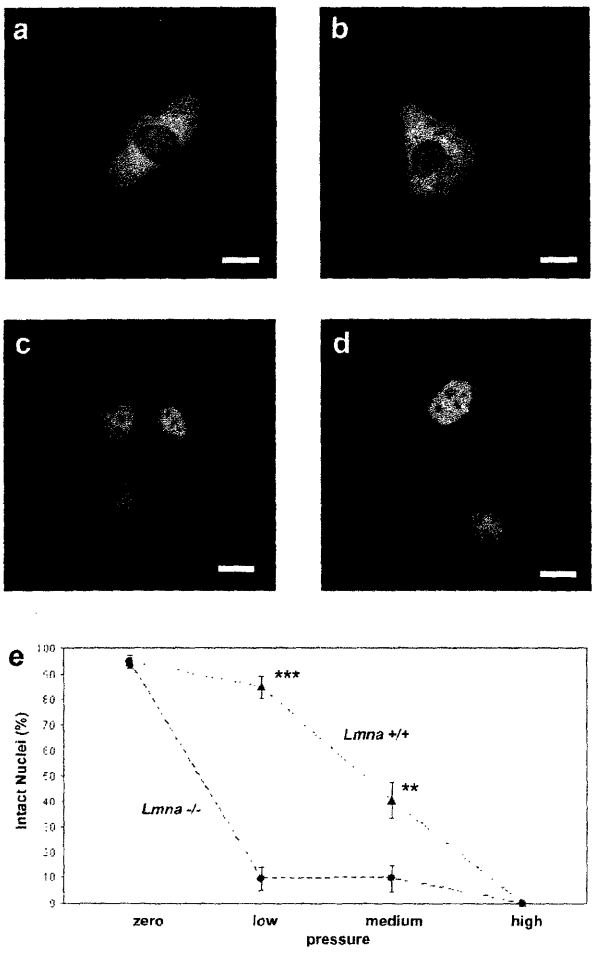


Figure 6.3. Nuclear fragility is increased in lamin A/C deficient cells. (a,b) Fluorescently labeled 70 kDa dextran is excluded from the nucleus following cytoplasmic injection, indicating an intact nuclear membrane in wild type (a) and *Lmna*$^{-/-}$ cells (b) under resting conditions. Scale bar 20 μm. (c) Nuclear injection at low pressure results in fluorescently labeled dextran contained in the nucleus of wild type cells, indicating that the nuclear integrity is preserved during injection. Scale bar 20 mm. (d) In contrast, nuclear integrity is disrupted in nuclei of *Lmna*$^{-/-}$ cells even at low pressure, leading to fluorescently labeled dextran escaping into the cytoplasm during injection. Scale bar 20 μm. (e) Nuclear rupture as a function of increasing injection pressure of dextran microinjection into the nucleus. Zero pressure: cytoplasmic injection at 500 hPa. Low pressure: nuclear injection at 10-20 hPa. (***: 84.7 ± 4.24% vs. 9.5 ± 4.53% intact nuclei for wild type and

Lmna$^{-/-}$ cells respectively, $p < 0.0001$, $n = 72, 42$). Medium pressure: nuclear injection at 100-500 hPa (**: 40.4 ± 7.16% vs. 9.5 ± 4.53% intact nuclei for wild type and *Lmna*$^{-/-}$ cells respectively, $p < 0.01$, $n = 47, 31$). High pressure: nuclear injection at 1,500 hPa, all cells showed nuclear rupture.

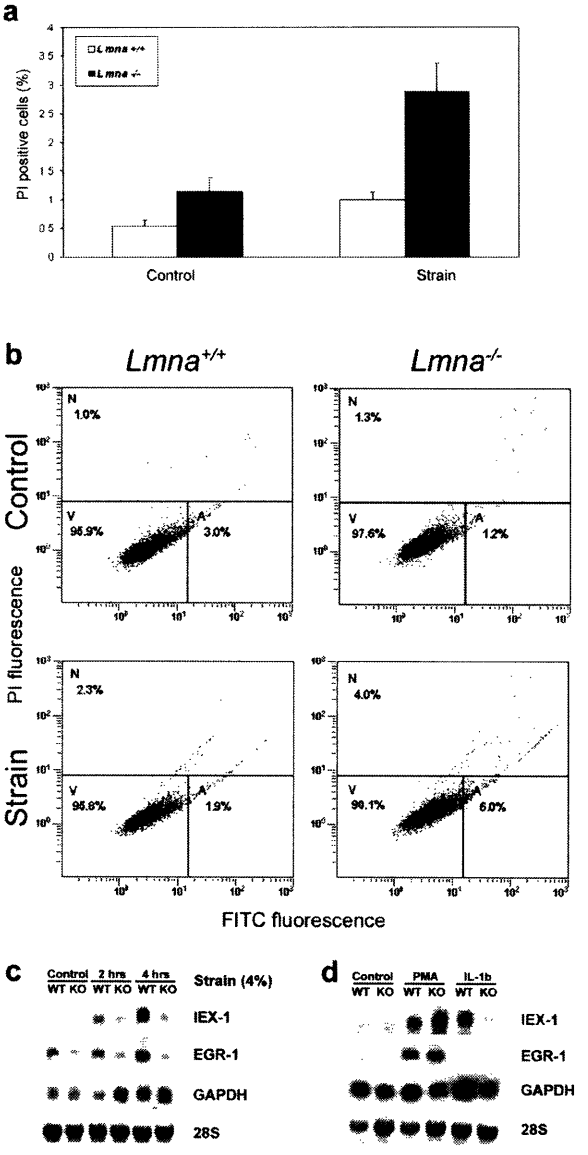


Figure 6.4. Impaired mechanotransduction in lamin A/C deficient cells. (a) *Lmna*$^{-/-}$ fibroblasts exhibited a significantly higher percentage of propidium iodide positive cells compared to wild type cells (2.88 ± 0.49 vs. 1.14 ± 0.14, $p < 0.01$, n = 7, 8) after 24 h strain application (10%, 1 Hz). Differences in unstrained cells were not significant (n = 9, 10). (b) Dual labeling with FITC conjugated annexin-V and propidium iodide uptake indicated that apoptotic (A) and necrotic (N) cell fractions are increased in *Lmna*$^{-/-}$ cells following prolonged strain. Viable cells (V) are propidium iodide and FITC negative.

(top left) wild type unstrained control. (top right) *Lmna*$^{-/-}$ unstrained control. (bottom left), wild type cells after 10% strain for 24 hours. (bottom right), *Lmna*$^{-/-}$ cells after 10% strain for 24 hours. (c) *Lmna*$^{-/-}$ (KO) fibroblasts exhibited attenuated mechanical induction of EGR-1 and IEX-1 at 2 and 4 hours of strain (4%) compared to *Lmna*$^{+/+}$ cells (WT). Expression of GAPDH was not negatively affected. (d) Cytokine (IL-1β) induced expression of IEX-1 was impaired in *Lmna*$^{-/-}$ cells (KO), while PMA-responsiveness remained intact in *Lmna*$^{-/-}$ cells.

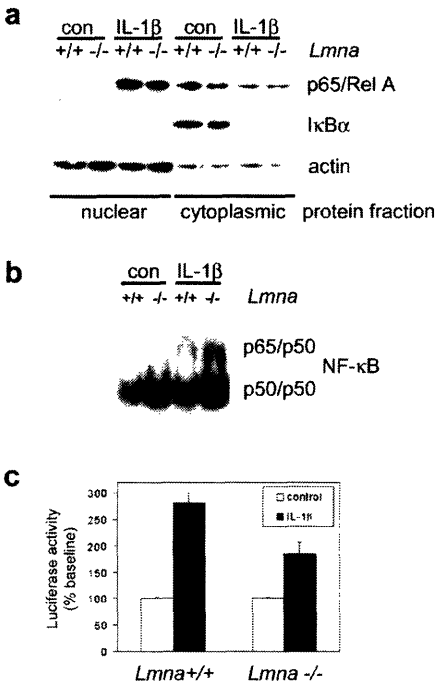


Figure 6.5. Defective NF-κB signaling in lamin A/C deficient cells. (a) Western analysis of nuclear and cytoplasmic protein fractions. Cytokine induced nuclear translocation of p65/RelA and degradation of cytoplasmic IκBα was indistinguishable between wild type and *Lmna*$^{-/-}$ fibroblasts. Cytoplasmic proteins were loaded at a lower concentration. (b) Electromobility shift assay for NF-κB target sequence using protein from the same nuclear fractions as in (a). Probe specificity was confirmed using unlabeled competitive and non-competitive probes. Identity of NF-κB subunits p50 and p65 was confirmed by supershift assay. (c) Cytokine-induced NF-κB regulated luciferase activity was significantly impaired in *Lmna*$^{-/-}$ cells (per cent baseline: 282 ± 18.5 vs. 185 ± 22.2, $p<0.001$, $n = 9$). Baseline activity was not significantly different between wild type and *Lmna*$^{-/-}$ cells.

6.5 References

1. Liu, J., Ben-Shahar, T.R., Riemer, D., Treinin, M., Spann, P., Weber, K., Fire, A., and Gruenbaum, Y. 2000. Essential roles for Caenorhabditis elegans lamin gene in nuclear organization, cell cycle progression, and spatial organization of nuclear pore complexes. *Mol Biol Cell* 11:3937-3947.
2. Hutchison, C.J. 2002. Lamins: building blocks or regulators of gene expression? *Nat Rev Mol Cell Biol* 3:848-858.
3. Holt, I., Ostlund, C., Stewart, C.L., Man Nt, N., Worman, H.J., and Morris, G.E. 2003. Effect of pathogenic mis-sense mutations in lamin A on its interaction with emerin in vivo. *J Cell Sci* 116:3027-3035.
4. Sullivan, T., Escalante-Alcalde, D., Bhatt, H., Anver, M., Bhat, N., Nagashima, K., Stewart, C.L., and Burke, B. 1999. Loss of A-type lamin expression compromises nuclear envelope integrity leading to muscular dystrophy. *J Cell Biol* 147:913-920.
5. Biamonti, G., Giacca, M., Perini, G., Contreas, G., Zentilin, L., Weighardt, F., Guerra, M., Della Valle, G., Saccone, S., Riva, S., et al. 1992. The gene for a novel human lamin maps at a highly transcribed locus of chromosome 19 which replicates at the onset of S-phase. *Mol Cell Biol* 12:3499-3506.
6. Lin, F., and Worman, H.J. 1993. Structural organization of the human gene encoding nuclear lamin A and nuclear lamin C. *J Biol Chem* 268:16321-16326.
7. Furukawa, K., Inagaki, H., and Hotta, Y. 1994. Identification and cloning of an mRNA coding for a germ cell-specific A-type lamin in mice. *Exp Cell Res* 212:426-430.
8. Lin, F., and Worman, H.J. 1995. Structural organization of the human gene (LMNB1) encoding nuclear lamin B1. *Genomics* 27:230-236.
9. Machiels, B.M., Zorenc, A.H., Endert, J.M., Kuijpers, H.J., van Eys, G.J., Ramaekers, F.C., and Broers, J.L. 1996. An alternative splicing product of the lamin A/C gene lacks exon 10. *J Biol Chem* 271:9249-9253.
10. Harborth, J., Elbashir, S.M., Bechert, K., Tuschl, T., and Weber, K. 2001. Identification of essential genes in cultured mammalian cells using small interfering RNAs. *J Cell Sci* 114:4557-4565.
11. Bonne, G., Di Barletta, M.R., Varnous, S., Becane, H.M., Hammouda, E.H., Merlini, L., Muntoni, F., Greenberg, C.R., Gary, F., Urtizberea, J.A., et al. 1999. Mutations in the gene encoding lamin A/C cause autosomal dominant Emery-Dreifuss muscular dystrophy. *Nat Genet* 21:285-288.
12. Fatkin, D., MacRae, C., Sasaki, T., Wolff, M.R., Porcu, M., Frenneaux, M., Atherton, J., Vidaillet, H.J., Jr., Spudich, S., De Girolami, U., et al. 1999. Missense mutations in the rod domain of the lamin A/C gene as causes of dilated cardiomyopathy and conduction-system disease. *N Engl J Med* 341:1715-1724.
13. Cao, H., and Hegele, R.A. 2000. Nuclear lamin A/C R482Q mutation in canadian kindreds with Dunnigan-type familial partial lipodystrophy. *Hum Mol Genet* 9:109-112.
14. Shackleton, S., Lloyd, D.J., Jackson, S.N., Evans, R., Niermeijer, M.F., Singh, B.M., Schmidt, H., Brabant, G., Kumar, S., Durrington, P.N., et al. 2000. LMNA, encoding lamin A/C, is mutated in partial lipodystrophy. *Nat Genet* 24:153-156.

15. Burke, B., and Stewart, C.L. 2002. Life at the edge: the nuclear envelope and human disease. *Nat Rev Mol Cell Biol* 3:575-585.
16. De Sandre-Giovannoli, A., Bernard, R., Cau, P., Navarro, C., Amiel, J., Boccaccio, I., Lyonnet, S., Stewart, C.L., Munnich, A., Le Merrer, M., et al. 2003. Lamin A Truncation in Hutchinson-Gilford Progeria. *Science.*
17. Eriksson, M., Brown, W.T., Gordon, L.B., Glynn, M.W., Singer, J., Scott, L., Erdos, M.R., Robbins, C.M., Moses, T.Y., Berglund, P., et al. 2003. Recurrent de novo point mutations in lamin A cause Hutchinson-Gilford progeria syndrome. *Nature* 423:293-298.
18. Raharjo, W.H., Enarson, P., Sullivan, T., Stewart, C.L., and Burke, B. 2001. Nuclear envelope defects associated with LMNA mutations cause dilated cardiomyopathy and Emery-Dreifuss muscular dystrophy. *J Cell Sci* 114:4447-4457.
19. Vigouroux, C., Auclair, M., Dubosclard, E., Pouchelet, M., Capeau, J., Courvalin, J.C., and Buendia, B. 2001. Nuclear envelope disorganization in fibroblasts from lipodystrophic patients with heterozygous R482Q/W mutations in the lamin A/C gene. *J Cell Sci* 114:4459-4468.
20. Bausch, A.R., Ziemann, F., Boulbitch, A.A., Jacobson, K., and Sackmann, E. 1998. Local measurements of viscoelastic parameters of adherent cell surfaces by magnetic bead microrheometry. *Biophys J* 75:2038-2049.
21. Lammerding, J., Kazarov, A.R., Huang, H., Lee, R.T., and Hemler, M.E. 2003. Tetraspanin CD151 regulates alpha6beta1 integrin adhesion strengthening. *Proc Natl Acad Sci U S A* 100:7616-7621.
22. Cheng, G.C., Briggs, W.H., Gerson, D.S., Libby, P., Grodzinsky, A.J., Gray, M.L., and Lee, R.T. 1997. Mechanical strain tightly controls fibroblast growth factor-2 release from cultured human vascular smooth muscle cells. *Circ Res* 80:28-36.
23. De Keulenaer, G.W., Wang, Y., Feng, Y., Muangman, S., Yamamoto, K., Thompson, J.F., Turi, T.G., Landschutz, K., and Lee, R.T. 2002. Identification of IEX-1 as a biomechanically controlled nuclear factor-kappaB target gene that inhibits cardiomyocyte hypertrophy. *Circ Res* 90:690-696.
24. Remoli, M.E., Giacomini, E., Lutfalla, G., Dondi, E., Orefici, G., Battistini, A., Uze, G., Pellegrini, S., and Coccia, E.M. 2002. Selective expression of type I IFN genes in human dendritic cells infected with Mycobacterium tuberculosis. *J Immunol* 169:366-374.
25. Caille, N., Tardy, Y., and Meister, J.J. 1998. Assessment of strain field in endothelial cells subjected to uniaxial deformation of their substrate. *Ann Biomed Eng* 26:409-416.
26. Inoh, H., Ishiguro, N., Sawazaki, S., Amma, H., Miyazu, M., Iwata, H., Sokabe, M., and Naruse, K. 2002. Uni-axial cyclic stretch induces the activation of transcription factor nuclear factor kappaB in human fibroblast cells. *Faseb J* 16:405-407.
27. Granet, C., Boutahar, N., Vico, L., Alexandre, C., and Lafage-Proust, M.H. 2001. MAPK and SRC-kinases control EGR-1 and NF-kappa B inductions by changes in mechanical environment in osteoblasts. *Biochem Biophys Res Commun* 284:622-631.

28. Caille, N., Thoumine, O., Tardy, Y., and Meister, J.J. 2002. Contribution of the nucleus to the mechanical properties of endothelial cells. *J Biomech* 35:177-187.
29. Markiewicz, E., Venables, R., Mauricio Alvarez, R., Quinlan, R., Dorobek, M., Hausmanowa-Petrucewicz, I., and Hutchison, C. 2002. Increased solubility of lamins and redistribution of lamin C in X-linked Emery-Dreifuss muscular dystrophy fibroblasts. *J Struct Biol* 140:241-253.
30. Bloom, S., Lockard, V.G., and Bloom, M. 1996. Intermediate filament-mediated stretch-induced changes in chromatin: a hypothesis for growth initiation in cardiac myocytes. *J Mol Cell Cardiol* 28:2123-2127.
31. Maniotis, A.J., Chen, C.S., and Ingber, D.E. 1997. Demonstration of mechanical connections between integrins, cytoskeletal filaments, and nucleoplasm that stabilize nuclear structure. *Proc Natl Acad Sci U S A* 94:849-854.
32. Moir, R.D., Yoon, M., Khuon, S., and Goldman, R.D. 2000. Nuclear lamins A and B1: different pathways of assembly during nuclear envelope formation in living cells. *J Cell Biol* 151:1155-1168.
33. Dreuillet, C., Tillit, J., Kress, M., and Ernoult-Lange, M. 2002. In vivo and in vitro interaction between human transcription factor MOK2 and nuclear lamin A/C. *Nucleic Acids Res* 30:4634-4642.
34. Lloyd, D.J., Trembath, R.C., and Shackleton, S. 2002. A novel interaction between lamin A and SREBP1: implications for partial lipodystrophy and other laminopathies. *Hum Mol Genet* 11:769-777.
35. Kumaran, R.I., Muralikrishna, B., and Parnaik, V.K. 2002. Lamin A/C speckles mediate spatial organization of splicing factor compartments and RNA polymerase II transcription. *J Cell Biol* 159:783-793.
36. Spann, T.P., Goldman, A.E., Wang, C., Huang, S., and Goldman, R.D. 2002. Alteration of nuclear lamin organization inhibits RNA polymerase II-dependent transcription. *J Cell Biol* 156:603-608.
37. Rao, L., Perez, D., and White, E. 1996. Lamin proteolysis facilitates nuclear events during apoptosis. *J Cell Biol* 135:1441-1455.
38. Kumar, A., and Boriek, A.M. 2003. Mechanical stress activates the nuclear factor-kappaB pathway in skeletal muscle fibers: a possible role in Duchenne muscular dystrophy. *Faseb J* 17:386-396.
39. Östlund, C., Bonne, G., Schwartz, K., and Worman, H.J. 2001. Properties of lamin A mutants found in Emery-Dreifuss muscular dystrophy, cardiomyopathy and Dunnigan-type partial lipodystrophy. *J Cell Sci* 114:4435-4445.

7 Conclusions and outlook

Measurements of subcellular biomechanics provide important insights into cellular function and present an important tool to study mechanotransduction events in specific cellular domains. Our experiments on extracellular force transduction through integrin receptors (Chapter 3) demonstrate the ability of magnetic bead experiments to quantitatively measure cell adhesion strength and dynamic adhesion strengthening. While we specifically examined the function of the C-terminal domain of tetraspanin CD151, these methods can generally be applied to a wide range of molecules involved in focal adhesion complex formation, enabling precise measurements of dynamic adhesion strengthening as a function of specific adhesion receptors or receptor associated proteins.

Magnetic bead experiments also provide a valuable tool for the study of cytoskeletal mechanics as demonstrated in Chapters 4 and 5. However, it is important to note that observed bead displacements depend not only on cytoskeletal stiffness but also on bead attachment strength, attachment angle, cell geometry, and, in the case of adult cardiac myocytes, bulk cell movement. Experiments analyzing the induced displacement of polystyrene beads on the cell surface or that directly measure cytoskeletal deformations using cross-correlation algorithm can circumvent some of these problems since the induced cytoskeletal deformations only depend on the applied force and are independent of magnetic bead movement. However, when comparing measurements between differently coated beads or different cell types, care has to be taken to keep experimental parameters as constant as possible. An additional limitation of our magnetic bead experiments is that bead placement on the cell is a random process and not actively controlled. While these factors are not of great concern when a large number of cells are available for experiments, they lead to more severe limitations when studying primary adult mouse cardiac myocytes which are extremely sensitive and short-lived. Despite these limitations, the magnetic bead experiments performed on adult mouse cardiac myocytes revealed significant changes in transverse passive cytoskeletal mechanics in myocytes from R120G-CryAB mutant mice, indicating adverse consequences of sarcomeric disarray and cytoplasmic desmin or αB-crystallin aggregates on cytoskeletal

mechanics. For these experiments, atomic force microscopy (AFM) can provide a valuable complement to measure transverse mechanical properties, as AFM can be applied to measure localized effects of the protein aggregates on cellular mechanics.

Magnetic bead experiments were also used to measure nuclear mechanics, but we found that the applied force was generally not sufficient to achieve reproducible and detectable nuclear deformations. In contrast, biaxial strain application to adherent cells yielded highly reproducible results on nuclear mechanics and revealed significantly impaired nuclear mechanics in lamin A/C deficient cells (see Chapter 6). Strain experiments were complemented by microinjection experiments to probe the rupture strength of the nuclear envelope, confirming the importance of lamin A/C for nuclear envelope integrity. While the experiments presented here were performed in fibroblasts that completely lack lamin A and lamin C, most human laminopathies arise from heterozygous lamin mutations. In addition, laminopathies can be caused by mutations in emerin (1-3) or the pre-lamin A/C processing enzyme Zmpste (4). Emerin is a ubiquitous 34-kDA nuclear envelope protein encoded by the STA gene that is located on the X-chromosome and strongly expressed in skeletal and cardiac muscle (5). X-linked Emery-Dreifuss muscular dystrophy is neuromuscular disease characterized by progressive skeletal muscle weakness and wasting, early contractures of the elbows, post-cervical muscles, and Achilles tendons, and cardiomyopathy. Emerin can directly bind to lamin A/C (6) and to chromatin through binding to the barrier-to-auto-integration factor protein BAF (7). We have here presented a first approach to develop independent tests for measuring structural and gene-regulatory functions of lamin A/C and other nuclear envelope proteins, thereby helping to clarify the effects of specific mutations. Preliminary experiments on emerin deficient primary mouse embryo fibroblasts found that emerin deficient cells had apparently normal nuclear mechanics (Fig. 7.1) but impaired activation of mechanosensitive genes in response to strain, indicating that emerin mutations do not primarily cause damage through impaired nuclear mechanics but through altered cellular signaling.

Additional experiments can be designed to investigate the effects of nuclear envelope protein mutations on cell mechanics and gene regulation in more details. These techniques will include three-dimensional imaging of cells and nuclear structure *in situ* in tissue affected by laminopathies, such as the heart, to observe the changes in cytoskeletal and nuclear structure during the process of the disease. Furthermore, cells derived from different tissues from laminopathy patients as well as from mice bearing specific lamin mutations can be cultured and subjected to strain and microinjection experiments to study the effect of specific mutations on nuclear stability and fragility, and nuclei isolated from these cells can be studied using AFM to examine nuclear mechanics more directly. Depending on the force application mode, these methods can reveal new insights on chromatin structure as well as the transmission of forces through the nuclear envelope to the nucleus. Applying these techniques to cells derived from different tissues and different mutations in humans and mice may then provide insight into why lamin A/C mutations cause diverse phenotypes.

In conclusion, cardiac myocytes and other cells react to diverse mechanical demands with a multitude of transient and long term responses to normalize the cellular mechanical environment. Several stretch-activated signaling pathways have been identified with multiple levels of cross-talk between pathways, and promising candidates for the cellular mechanosensor, the site of signal transduction initiation, are now emerging. A more detailed insight into the mechanism of mechanotransduction is expected from the combination of experimental and theoretical approaches to study mechanotransduction at molecular, cellular, and systemic levels. Computational approaches that simulate conformational changes in response to mechanical forces using molecular dynamics as well as experiments on molecular mechanics will assist in the understanding of mechanotransduction on a molecular level. The creation of genetically engineered mice has greatly facilitated the understanding of individual protein function in mechanically-induced signaling. Creation of other transgenic species will further improve our understanding mechanotransduction in hearts closer resembling the human heart. In addition, improved imaging techniques allow direct insights into the activation and translocation of signaling molecules in live cells.(8) The combination of studying

mechanically-induced signaling in single cells together with the validation of these findings in the whole organism should lead to significant advances in the understanding of mechanotransduction in the next years. Along with the experimental and theoretical study of molecular mechanics and conformational changes in response to applied forces, these emerging techniques will also address the need for a better fundamental understanding of the mechanisms of initiating the signal transduction response and help in the identification of the elusive cellular mechanosensor(s).

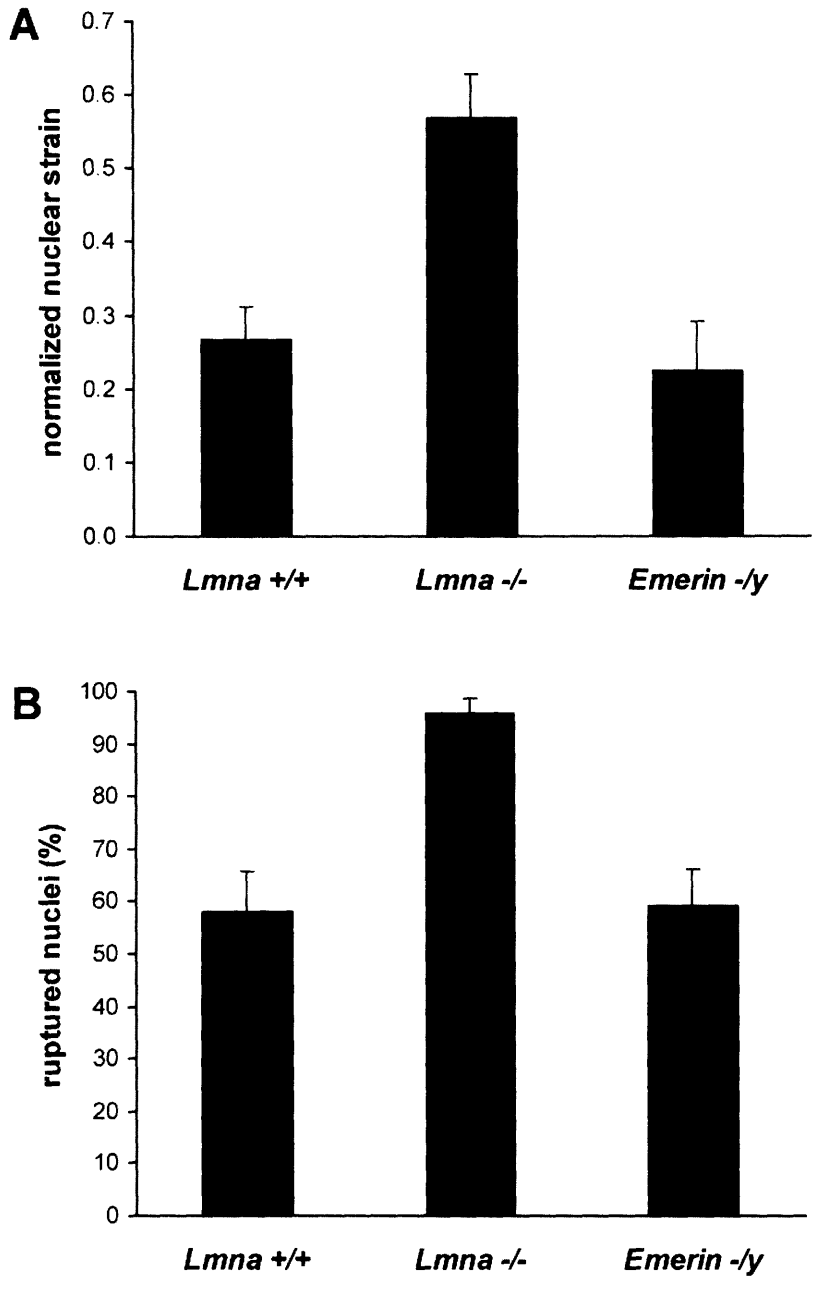


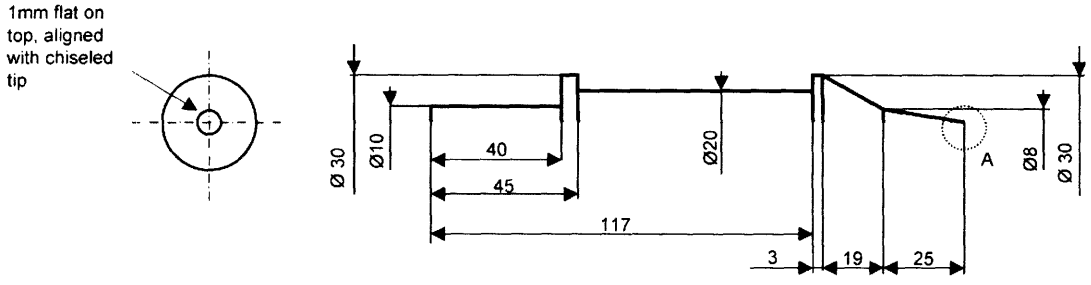
Figure 7.1. Nuclear mechanics in wild-type, emerin deficient and lamin A/C deficient primary mouse embryo fibroblasts. (A) Lamin A/C deficient fibroblasts have significantly increased nuclear deformations under biaxial strain. Emerin deficient cells are comparable to wild-type. (B) Nuclear microinjection experiments indicate increased nuclear fragility in lamin A/C deficient but not in emerin deficient cells.

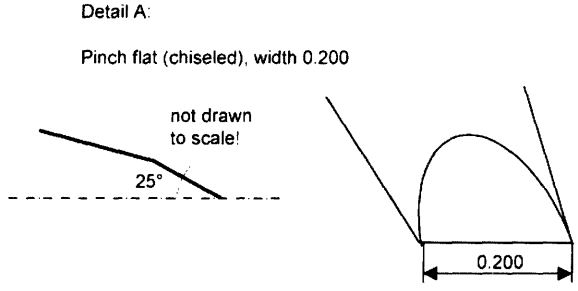
7.1 References

1. Bione, S., Maestrini, E., Rivella, S., Mancini, M., Regis, S., Romeo, G. & Toniolo, D. (1994) *Nat Genet* **8,** 323-7.
2. Ostlund, C. & Worman, H. J. (2003) *Muscle Nerve* **27,** 393-406.
3. Funakoshi, M., Tsuchiya, Y. & Arahata, K. (1999) *Neuromuscul Disord* **9,** 108-14.
4. Bergo, M. O., Gavino, B., Ross, J., Schmidt, W. K., Hong, C., Kendall, L. V., Mohr, A., Meta, M., Genant, H., Jiang, Y., Wisner, E. R., Van Bruggen, N., Carano, R. A., Michaelis, S., Griffey, S. M. & Young, S. G. (2002) *Proc Natl Acad Sci U S A* **99,** 13049-54.
5. Lattanzi, G., Ognibene, A., Sabatelli, P., Capanni, C., Toniolo, D., Columbaro, M., Santi, S., Riccio, M., Merlini, L., Maraldi, N. M. & Squarzoni, S. (2000) *Differentiation* **66,** 208-17.
6. Sakaki, M., Koike, H., Takahashi, N., Sasagawa, N., Tomioka, S., Arahata, K. & Ishiura, S. (2001) *J Biochem (Tokyo)* **129,** 321-7.
7. Haraguchi, T., Koujin, T., Segura-Totten, M., Lee, K. K., Matsuoka, Y., Yoneda, Y., Wilson, K. L. & Hiraoka, Y. (2001) *J Cell Sci* **114,** 4575-85.
8. Weijer, C. J. (2003) *Science* **300,** 96-100.

Appendix A. Magnetic trap design draft

Design drafts for magnetic trap core. The pole material is CMI-C steel, which has a relatively high permeability and saturation. Note that the tip is not axis-symmetric, it is chiseled to a straight edge, see *Detail A*. All units are given in <mm>. Following machining, the part is annealed to improve magnetic properties and improve corrosion resistance. Some of our magnetic traps were additionally nickel-plated. Wire is wrapped around the central part of the core and secured using epoxy.





Appendix B. Nuclear strain as an indicator of nuclear mechanical properties

Cellular strain measurements can be used to infer to the relative stiffness of the nucleus compared to the cytoskeleton. Figure B.1*A* depicts a simplified schematic of an adherent cell under strain. The cytoskeleton is firmly attached to the elastic membrane through focal adhesion sites predominantly located at the cell periphery. The nucleus is physically connected to the cytoskeleton through nuclear/cytoskeletal interactions that are not yet fully established. A one-dimensional model of the simplified cellular mechanics is given in figure B.1*B*, where the spring constants c_{cyto} and c_{nucl} represent the cytoskeletal and nuclear elasticity respectively and Δx is the total applied strain or displacement. For a combination of springs in series, the force is constant throughout the system. Since the force of each spring under tension is $F_i = c_i \Delta x_i$, it follows that

$$F = F_i = c_{cyto} \Delta x_{cyto} = c_{nucl} \Delta x_{nucl} \tag{1}$$

or

$$\Delta x_{cyto} = \frac{c_{cyto}}{c_{nucl}} \Delta x_{nucl} \tag{2}$$

For the combination of springs in series, the total deformation Δx can be expressed as the sum of the individual deformations, i.e.

$$\Delta x = \Delta x_{cyto} + \Delta x_{nucl} + \Delta x_{cyto} \tag{3}$$

Combining equations (2) and (3) yields the following expressions for the nuclear and cytoskeletal contribution to the total deformation

$$\begin{aligned} \frac{\Delta x_{cyto}}{\Delta x} &= \frac{c_{nucl}}{2c_{cyto} + c_{nucl}} \\ \frac{\Delta x_{nucl}}{\Delta x} &= \frac{c_{cyto}}{2c_{cyto} + c_{nucl}} \end{aligned} \tag{4}$$

In the normal case of a significantly stiffer nucleus (c_{nucl} ~ 5-10 times c_{cyto}), the cytoskeletal deformation closely matches the applied membrane strain, while the nuclear deformation is significantly smaller. Impaired nuclear mechanics on the other hand will lead to significantly larger nuclear deformation under strain (compare with chapter 6).

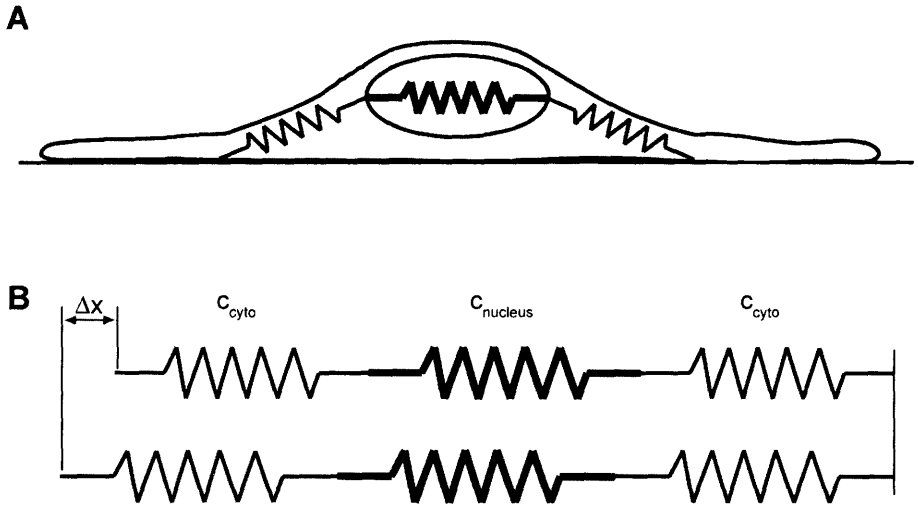


Figure B.1. Schematic of a cell subjected to membrane strain. (*A*) Membrane strain is transmitted to the cytoskeleton through focal adhesion sites located at the cell periphery. Forces are transmitted through the cytoskeletal network to the centrally located nucleus. Springs represent the elastic contribution of the cytoskeleton and the nucleus respectively. (*B*) Linearized one-dimensional model of the cellular mechanics. Cytoskeleton and nucleus are represented by springs in series, and the applied strain is the sum of the individual spring length changes.

Appendix C. Monocular 3-D magnetic bead microrheometry*

A novel single-lens three-dimensional microscope imaging technique is proposed to quantitatively detect positions and movements of micro-beads that are bound to the cell membrane or embedded in the cytoskeleton and manipulated by magnetic tweezers (Fig. 1). The technique is used to explore cellular mechanisms of mechanotransduction in cardiac hypertrophy, i.e. the biochemical response to mechanical stimulation. The measured relationship between force (stress) and three-dimensional displacement (strain) yields new insights into local cellular mechanics such as active force generation, passive elastic properties, and the mechanical coupling between the cytoskeleton and the extracellular matrix. The main goal of the present research is to characterize changes in mechanical properties that are associated with eccentric or concentric hypertrophy and to identify how these changes might activate mechanotransduction pathways that lead to different types of hypertrophy.

The proposed 3-D microscope imaging system contains either a relay lens system with an off-axis rotating aperture placed between the imaging lens and the CCD camera (Fig.2) or an off-axis rotating aperture located in between the field and the aperture diaphragms of the Köhler illumination system of the microscope (Fig.3). In both cases, the rotating aperture allows adjustable non-equilateral spacing between images of out-of-focus object points to achieve higher spatial resolution, increased sensitivity and higher sub-pixel displacement accuracy.

By shifting the aperture off-axis we can have controlled sampling of the defocus blur whose diameter is proportional to the distance of the object point from the focal plane. Figure 4 illustrates the concept behind measuring out-of-plane coordinates of object points by sampling the defocus blur, with an off-axis rotating aperture, and measuring its diameter. The single aperture avoids overlapping of images from different object regions hence it increases the spatial resolution of the measurement. The rotating

* Work presented in this chapter was done with Janos Rohaly and Douglas Hart. This is an abstract that was presented at the 11th Intl. Symposium on Applications of Laser techniques to Fluid Mechanics Lisbon, Portugal, 2002. A patent for the system is pending with J Rohaly and DP Hart as the co-inventors.

aperture allows taking images at several aperture positions and this can be interpreted as having several cameras with different viewpoints, which generally increases measurement sensitivity. Tracking the depth related image disparities along the rotation of the aperture increases the sensitivity of detecting 3-D positions by several orders of magnitude. Alternatively, one can have only a few aperture positions and the related image disparities can be fed back to the system to reposition the aperture to achieve greater accuracies. The active rotating aperture solves the ambiguity problem of object location relative to the focal plane by imposing a 180 deg phase shift in between objects at the two sides of the focal plane. However, due to the degradation of image movement by aberrations (Fig.5) the best performance is achieved by imposing a slight defocus on the entire object field. Such an off-axis rotating aperture can be incorporated into a relay lens system in between the microscope objective and the image-recording device, as shown in Fig. 2.

The advantage of this solution is its simplicity, however, sampling the defocus spot by an off-axis rotating aperture in the imaging path inherently has its own limitations. The most important is that it requires stepping down the aperture, which reduces resolution and requires stronger illumination. Lens aberration related systematic bias in the created image disparity could also be a problem.

All of these barriers can be overcome by placing the off-axis rotating aperture in the illumination side of the microscope in between the field and the aperture diaphragms of the Köhler illumination system, which creates oblique illumination (Fig 3). Any movement of the off-axis rotating iris results in movement of all out-of-focus image points, as demonstrated in Fig.6. Although the applied off-axis rotating aperture gives weaker illumination intensity, and reduces the specimen size that can be imaged, the system is free of all resolution and aberration related limitations. This solution is also preferred if the objective has very long depth-of-focus, since this would create very small image disparities limiting the sensitivity of depth measurements.

The proposed monocular 3-D imaging technique was successfully applied on tracking micro-beads in the above-described application.

Figure C.1. A magnetic trap, mounted on a micromanipulator, creates the required force to manipulate the magnetic beads attached to single myocytes placed on a temperature-controlled microscope stage

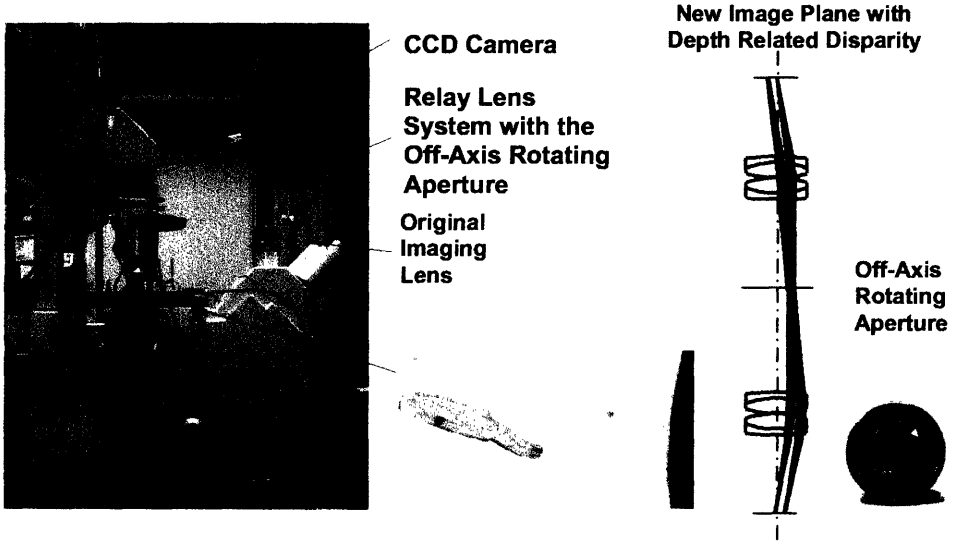


Figure C.2. A three-dimensional microscopes including a relay lens system with rotating aperture.

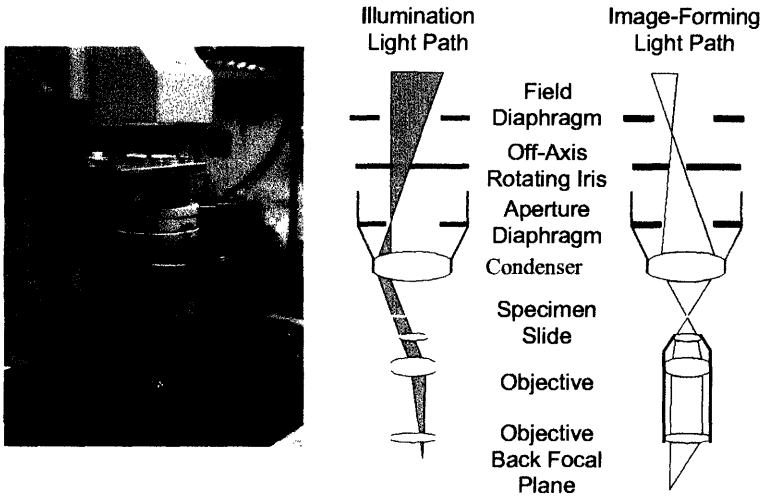


Figure C.3. Inverted light microscope with a rotating iris that creates adjustable oblique illumination

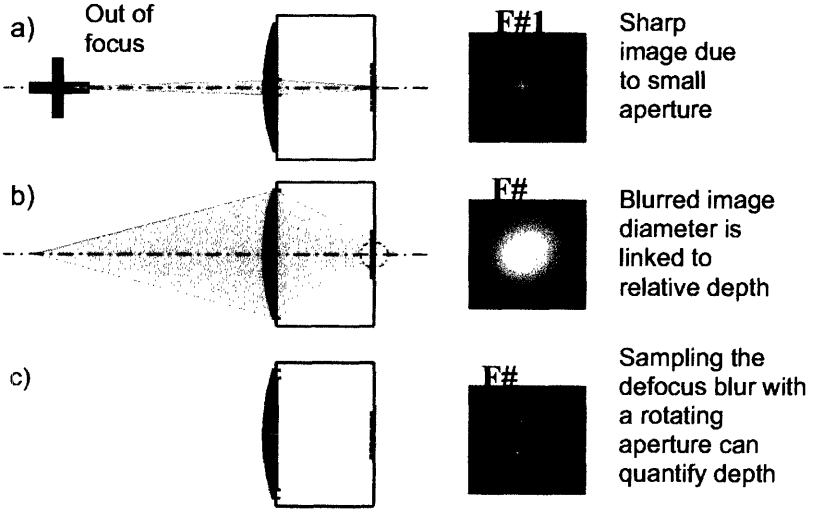


Figure C.4. A rotating off-axis aperture can sample the defocus blur to give quantitative depth information

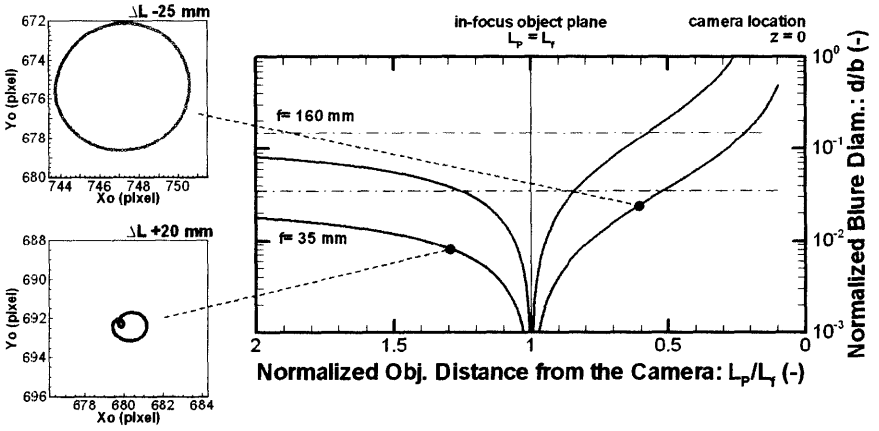


Figure C.5. Blur diameter of out-of-focus object points around the focal plane and the distortion of spot path created by lens aberration

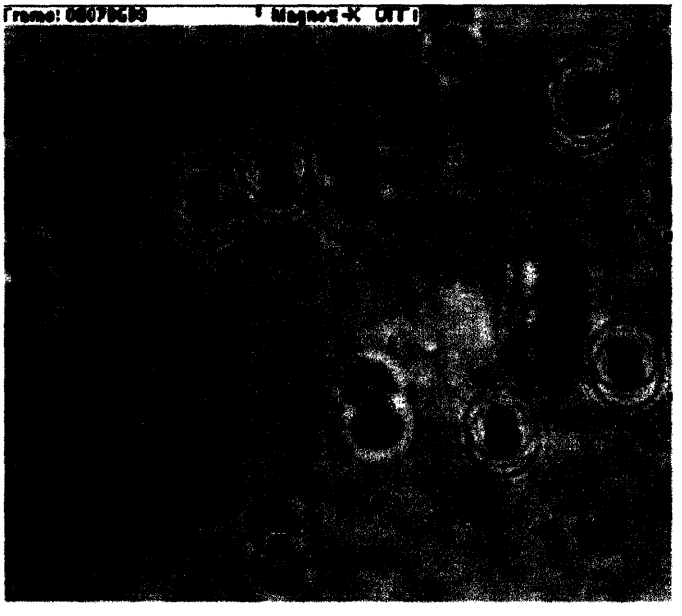


Figure C.6. The diameter of the circular image paths of micro-beads represents their z position. Suspension of 4.5μm magnetic beads, viewed through a 63x objective. Two images from a sequence are overlaid (non-constant focus).

Appendix D. Isolation of adult mouse cardiac myocytes

The following protocol was adapted from the procedure protocol (PP00000125) provided by the Alliance for Cellular Signaling (AfCS) at *http://www.signaling-gateway.org/.*

Procedure setup

1. Prepare perfusion buffers; myocyte digestion buffer (MC digestion buffer); myocyte stopping buffers (MC stop 1 and MC stop 2); myocyte plating medium (MC plating medium); myocyte culture medium (MC culture medium); and laminin-coated dishes (see "Laminin Coating of Culture Dishes" below) fresh daily. Myocyte plating medium stock is prepared in advance and supplemented with BDM on the day of experiments. Myocyte culture medium is prepared in advance and can be stored several weeks. Equilibrate MC plating medium and MC culture medium at 37 °C in a 5% CO2 incubator (for at least 2 hr to adjust temperature and pH). Sterilize instruments in autoclave.
2. Prepare the perfusion apparatus. Set the circulating water bath so that the outflow from the tip of the cannula is 36-37 °C. Check the flow rate of the pump and adjust to 3 ml/min.
3. Run ~30 ml of ethanol and then 100 ml of purified water through the perfusion system; then run perfusion buffer through the system for at least 5 min.
4. Add 10 ml of room temperature perfusion buffer to a 60-mm culture dish for heart collection. Add 20 ml of room temperature perfusion buffer to a 100-mm culture dish for heart cannulation.
5. Position the cannula with the tip close to the surface of perfusion buffer in the 100-mm dish.
6. Cut several small pieces (10 – 15 cm) of 6-0 surgical silk (Genzyme, Fall River, MA), knot loosely, and place on the adjustable stage (this will be used to secure the aorta to the cannula).

Removal and cannulation of the heart

7. Inject the mouse i.p. with 0.3 ml Heparin sodium salt solution (1000 U/ml, UCB Biomedicals). Wait for 20-30 min.
8. Anesthetize the mouse with i.p. injection of 0.3 ml pentobarbital cocktail, consisting of 0.8 ml Nembutal sodium solution (50 mg/ml, Abbott Laboratories), 2 ml ethyl alcohol (200 proof), and 8 ml saline (0.9% sodium chloride). When the mouse is anesthetized, it will lose consciousness and roll over on its side. Check with a toe pinch to ensure that the mouse is fully anesthetized. Transfer the mouse to the surgery/perfusion area.
9. Once the animal is on the surgery area, wipe the chest with 70% ethanol.
10. Using a dissecting microscope, open the peritoneal cavity and chest with small scissors and use forceps to peel back the rib cage to expose the heart. Lift the heart gently using forceps. Identify and cut the pulmonary vessels, which will make it easier to identify and cut the aorta. Cut the aorta at about 2-mm from its entry into the heart and immediately place the heart in a 60-mm dish containing 10 ml of perfusion buffer at room temperature. Too long a section of aorta will make the aorta harder to identify and lift onto the cannula. Conversely, too short a section of aorta will make it harder to tie off the aorta on the cannula and increase the likelihood of pushing the cannula through the aortic valve, preventing good perfusion.
11. Remove extraneous tissues (thymus and lungs), if necessary, and transfer heart to the 100-mm dish with perfusion buffer at room temperature. Re-start peristaltic pump with perfusion buffer (3 ml/min).
12. Cannulate the heart using fine-tip forceps to slide the aorta onto the cannula so that the tip of the cannula is just above the aortic valve (check the 1-mm notch on the cannula to ensure proper cannulation; see description of cannula below). Attach a small vascular clip at the end of the cannula to prevent the heart from falling. Start the perfusion immediately (3 ml/min). Tie the aorta to the cannula with 6-0 silk thread. Total time to cannulate the heart should be less than 1 min (in practice, this varied from 1-5 min for our mice).

Heart perfusion and enzyme digestion

13. Perfuse the heart with perfusion buffer for 4 min at 3 ml/min (this flushes blood from the vasculature and removes extracellular calcium to stop contractions).
14. Switch to the MC digestion buffer and perfuse for 8-10 min at 3 ml/min. If the heart is well perfused during the enzyme digestion, the heart will become swollen and turn slightly pale, and separation of muscle fibers on the surface of the heart may become apparent.

Myocyte dissociation

15. Once enzyme digestion of the heart is complete (heart appears swollen, pale, and flaccid), cut the heart from the cannula just below the atria using sterile, fine scissors. Place the ventricles in a 60-mm dish containing 2.5 ml of MC digestion buffer.
16. Remove atria and right ventricle. Cut the heart in half and begin to gently tease the ventricle into several small pieces with fine forceps. Pipette gently several times with a sterile plastic transfer pipette (~2-mm opening). This process takes 60 to 90 sec. The tissue should be very flaccid, almost falling apart on its own, and require very little force to pull apart, which will indicate a good digestion.
17. Transfer the cell suspension to a 15-ml polypropylene conical tube. Rinse the plate with 2.5 ml of room temperature myocyte stopping buffer 1 (MC stop 1), and combine with the cell suspension for a final volume of 5 ml. (Note: MC stop 1 contains serum to inactivate proteases; the final calf serum concentration is 5%, and the final calcium concentration is 12.5 μM).
18. Continue to dissociate the heart tissue gently, using sterile plastic pipettes with different sized openings (5 ml pipette and then 2 ml pipette), until all the large pieces of heart tissue are dispersed in the cell suspension. Avoid vigorous agitation to minimize shearing of the cells. This process should take 3 to 5 min.
19. Transfer cells into a sterile 15-ml tube. From this point forward, sterile techniques should be maintained, and all subsequent steps are performed under a laminar flow culture hood. (All steps conducted at room temperature.)

Calcium reintroduction

20. Allow the myocytes to sediment by gravity for 8 to 10 min in the 15-ml tube.
21. Carefully remove supernatant (which contains mostly dead cells). Resuspend the new pellet in 10 ml of room temperature myocyte stopping buffer 2 (MC stop 2). (Note: MC stop 2 contains 5% serum, and the final calcium concentration is 12.5 μM).
22. Transfer the cells (in 10 ml) to a 60-mm non-treated polystyrene dish. To evenly distribute the myocytes in the dish, move the dish forward and backward and side to side, but do not swirl the dish. Myocytes in the 60-mm dish can be monitored under a microscope during calcium reintroduction. If during the reintroduction period, the myocytes begin to round up, or become very granular, the myocyte quality might be poor. A decision may be made as to whether to continue with the prep or to start over. A single heart should yield ~ 1 million myocytes, with at least 60% rod shaped. In our experience, we obtained more than 80% rod-shaped cells from wild-type mice. Isolation preps with <60% rod-shaped cells were discarded. For mutant mice (R120G-CryAB or D7-des) however, viability was significantly reduces, with generally only 10-50% of viable cells. Preps with <10% were discarded for mutant mice.
23. Add 50 μl of calcium chloride, 10 mM (10 mM CaCl2); final concentration is increased to 62 μM. Mix well and incubate for 4 min at room temperature.
24. Add 50 μl of 10 mM CaCl2; final concentration is increased to 112 μM. Mix well and incubate for 4 min at room temperature.
25. Add 100 μl of 10 mM CaCl2; final concentration is increased to 212 μM. Mix well and incubate for 4 min at room temperature.
26. Add 30 μl of calcium chloride, 100 mM (100 mM CaCl2); final concentration is increased to approximately 500 μM. Mix well and incubate for 4 min at room temperature.
27. Add 50 μl of 100 mM CaCl2; final concentration is increased to approximately 1 mM. Mix well and incubate for 4 min at room temperature.
28. Transfer the myocytes to a new 15-ml tube and allow the myocytes to sediment by gravity (8 to 10 min).

29. Remove supernatant and resuspend pellets in 5-10 ml (depending on cell yield) of MC plating medium (1.2 mM Ca2+) at 37 °C.

30. Transfer ~0.5-1 ml of cell suspension into 35-mm polystyrene dish containing 2 ml of MC plating medium. Inspect cell number, for bead experiments, the dish should contain large number of single cells with approximately 50-100 μm between cells.

Plating myocytes, myocyte attachment, and culture

31. Prepare 5-10 laminin-coated 35-mm cell culture dishes by washing 1x with PBS- and then adding 2 ml of MC plating medium per dish. MC plating medium should have been equilibrated for 2 to 3 hr at 37 °C in a 5% CO2 incubator. Make sure the myocytes are resuspended well by gently pipetting (use a 10-ml pipette). (Note: at this stage the myocytes are in MC plating medium, which contains butanedione monoxime [BDM], a contraction inhibitor [also found in the perfusion buffer], and 5% calf serum.)

32. Plate the appropriate amount of rodshaped myocytes in desired vessels: ~0.5-2 ml (containing ~50,000 rod-shaped myocytes) in a laminin-coated 35-mm dish; (see comments above) Use a 5-ml pipette to plate two 35-mm dishes at once to prevent myocytes from settling in the pipette. During the plating procedure, resuspend the myocytes constantly to ensure they do not settle to the bottom of the tube, which will cause variation in plating density. Once the myocytes are plated, mix the myocytes in the dishes by gently sliding the tray forward and backward and side to side 3 to 4 times on the surface of the culture hood (in a cross-like pattern); never swirl the medium in the dish or the myocytes will not plate evenly. Place finished trays immediately in a 5% CO2 incubator at 37 °C. Incubate for 1 hr to allow myocyte attachment.

33. After 1 hr, aspirate the plating medium with a sterile Pasteur pipette into a vacuum flask. Wash each dish with approximately 1.5 ml MC culture medium to remove unattached myocytes and debris and aspirate the wash.

34. Add MC culture medium, which has been equilibrated at 37 °C in a 5% CO2 incubator for at least 2 to 3 hr, to the washed cells (3 ml medium to 35-mm

dishes). Immediately return myocytes to the incubator. (Note: at this stage, the myocytes are in MC culture medium, which contains 0.1 mg/ml bovine serum albumin [BSA], but no BDM or calf serum.). Cells are now ready for experiments.

Cleaning the perfusion rig

Clean the perfusion system by flushing the tubing with purified water. Next, fill the reservoirs and tubing with 70% ethanol and let the tubing soak in ethanol for at least 20 min. Lastly, run the ethanol through and wash the system twice with purified water.

Laminin coating of culture dishes

1. Thaw laminin (L-2020, Sigma) stock solution (aliquots of 100 μg/ml) on ice or in refrigerator prior to coating the plate.
2. Add 10 ml of ice-cold phosphate buffered saline (PBS), 1X (CaCl2/MgCl2-free) to the laminin stock for a final concentration of 10 μg/ml.
3. Add 1.5 ml of laminin coating solution to cover the bottom of 35-mm plates, and incubate at 4 °C overnight.
4. Remove laminin coating solution just prior to plating the myocytes.
5. Use the plates on the day they are prepared.

Reagents and materials

Cannula

The cannula is a very important part of the prep. It is a 20-g needle with the sharp tip cut off and the nub filed flat and smooth (this is important to prevent snagging the aorta when hanging). It is also useful to make 1-mm and 2-mm notches on the bottom of the cannula to determine how far the cannula is inside the aorta.

Perfusion systems

The perfusion system utilizes preheating of perfusion solutions just prior to entry into the hanging heart (see Figure 4 of the AfCS protocol). Perfusion solutions are made and kept in 50-ml tubes at room temperature until use. Changes in perfusion solutions are achieved by stopping the peristaltic pump and moving the inlet tube to the container with the next desired solution. The total volume of solution contained in the perfusion tubing (inlet to cannula) is 4 ml; this is taken into account for the volumes of perfusion given in the procedures. Perfusion solutions are pumped through a heat exchange coil where the temperature is raised to 37 °C for passage into the heart via the cannula. In practical terms, the temperature of the circulation water bath is set to about 39 °C so that the actual temperature of the heart during the perfusion is 37 °C. A picture of the perfusion system is shown in Figure 5 of the AfCS protocol.

Buffers (see protocols below)

Perfusion buffer: AfCS Solution Protocol PS00000451

Myocyte digestion buffer (MC digestion buffer): AfCS Solution Protocol PS00000447

Myocyte stopping buffer 1 (MC stop 1): AfCS Solution Protocol PS00000449

Myocyte stopping buffer 2 (MC stop 2): AfCS Solution Protocol PS00000450

Myocyte plating medium (MC plating medium): AfCS Solution Protocol PS00000448

Myocyte culture medium (MC culture medium): AfCS Solution Protocol PS00000446

Calcium chloride, 10 mM (10 mM CaCl2): AfCS Solution Protocol PS00000440

Calcium chloride, 100 mM (100 mM CaCl2): AfCS Solution Protocol PS00000441

Laminin stock solution (laminin stock): AfCS Solution Protocol PS00000444

Name: Stock Perfusion Buffer, 1X

Abbreviation: 1X Perf buf stock

Volume: 500 ml

Components:

Reagent	F.W. or Stock Conc.	Quantity	Final Conc.
Sodium chloride (NaCl)	54.8 g/ml	3.3 g	113 mM
Potassium chloride (KCl)	74.6 g/ml	0.175 g	4.7 mM
Potassium phosphate monobasic (KH_2PO_4)	136.1 g/mol	0.041 g	0.6 mM
Sodium phosphate dibasic (Na_2HPO_4)	142 g/mol	0.0425 g	0.6 mM
Magnesium sulfate heptahydrate ($MgSO_4$-$7H_2O$)	246.5 g/mol	0.15 g	1.2 mM
Phenol red	376.4 g/mol	0.006 g	0.032 mM
Sodium bicarbonate ($NaHCO_3$)	84 g/mol	0.505 g	12 mM
Potassium bicarbonate ($KHCO_3$)	101 g/mol	0.505 g	10 mM
HEPES ($C_8H_{18}N_2O_4S$)	238.3 g/mol	1.19 g	10 mM
Taurine	125.1 g/mol	1.875 g	30 mM

Preparation:

1. Add all components to 450 ml purified water in a 1-L beaker.
2. Stir until all components are thoroughly dissolved.
3. Transfer to a 500-ml graduated cylinder and adjust final volume to 500 ml with purified water.
4. Sterilize the solution by filtering through 0.2 μm filter unit
5. Store at 4 C for up to 1 week.

Name: Perfusion Buffer, pH 7.46

Abbreviation: Perf buf

Volume: 500 ml

Components:

Reagent	F.W. or Stock Conc.	Quantity	Final Conc.
Stock perfusion buffer	1X	490 ml	.98X
2,3-Butanedione monoxime (BDM)	500 mM	10 ml	10 mM
Glucose	180.2 g/mol	0.5 g	5.5 mM

Preparation:

1. Weigh glucose and add to 490 ml of stock perfusion buffer in a 500-ml beaker.
2. Add 10 ml of BDM to beaker and stir until all components are thoroughly dissolved.
3. Sterilize the solution by filtering through 0.2 μm filter unit.
4. Final pH is 7.46 after filtration.
5. Use solution on same day. Do not store!

Name: Myocyte stopping buffer 1

Abbreviation: MC stop 1

Volume: 10 ml

Components:

Reagent	F.W. or Stock Conc.	Quantity	Final Conc.
Perfusion buffer	1X	9 ml	1X
Bovine calf serum (BCS)	100%	1 ml	10%
Calcium chloride ($CaCl_2$)	10 mM	12.5 µl	12.5 µM

Preparation:

1. Pipette 9 ml of perfusion buffer into 50 ml tube, add 1 ml of BCS and 12.5 µl of $CaCl_2$. All reagents should be sterile, otherwise sterilize the solution by filtering through 0.2 µm filter unit.
2. Use solution on same day. Do not store!

Name: Myocyte stopping buffer 2

Abbreviation: MC stop 1

Volume: 10 ml

Components:

Reagent	F.W. or Stock Conc.	Quantity	Final Conc.
Perfusion buffer	1X	9.5 ml	1X
Bovine calf serum (BCS)	100%	0.5 ml	10%
Calcium chloride ($CaCl_2$)	10 mM	12.5 µl	12.5 µM

Preparation:

1. Pipette 9.5 ml of perfusion buffer into 50 ml tube, add 0.5 ml of BCS and 12.5 µl of $CaCl_2$. All reagents should be sterile, otherwise sterilize the solution by filtering through 0.2 µm filter unit.
2. Use solution on same day. Do not store!

Name: Myocyte digestion buffer

Abbreviation: MC digestion buffer

Volume: 50 ml

Components:

Reagent	F.W. or Stock Conc.	Quantity	Final Conc.
Perfusion buffer	1X	50 ml	1X
Liberase blendzyme 1 (Roche)	NA	12.5 mg	0.25 mg/ml
Trypsin 2.5% (10x, GIBCO)	25 mg/ml	0.278 ml	0.14 mg/ml
Calcium chloride ($CaCl_2$)	100 mM	6.25 µl	12.5 µM

Preparation:

1. Weigh liberase blendzyme and add to 50 ml of perfusion buffer in a 100-ml beaker.
2. Add trypsin and $CaCl_2$ to the beaker.
3. Stir until all components are thoroughly dissolved.
4. Sterilize the solution by filtering through 0.2 µm filter unit.
5. Do not store solution. Prepare fresh as needed.

Name: Myocyte plating medium

Abbreviation: MC plating medium

Volume: 500 ml

Components:

Reagent	F.W. or Stock Conc.	Quantity	Final Conc.
Minimum essential medium (MEM), 1X with Hanks' salts and L-glutamine (GIBCO)	1X	500 ml	0.9X
Bovine calf serum (BCS)	100%	27.5 ml	5%
Penicillin/streptomycin	100,000 U/ml	5 ml	100 U/ml
BDM	500 mM	10 ml	10 mM

Preparation:

1. Prepare stock solution by adding BCS and Pen/strep to MEM, keep everything sterile. This stock can be stored at 4°C.
2. On the day of experiments, pipette 49 ml of stock into 50-ml tube.
3. Add 1 ml of BDM solution (500 mM). Keep everything sterile.
4. Equilibrate final solution in incubator (5% CO_2) for 2-3 hours prior to use. Use on same day. Do not store!

Name: Myocyte culture medium

Abbreviation: MC culture medium

Volume: 500 ml

Components:

Reagent	F.W. or Stock Conc.	Quantity	Final Conc.
Minimum essential medium (MEM), 1X with Hanks' salts and L-glutamine (GIBCO)	1X	490 ml	1X
Bovine serum albumine	100 mg/ml	500 μl	0.1 mg/ml
Penicillin/streptomycin	100,000 U/ml	5 ml	100 U/ml
L-glutamine	200 mM	5 ml	2 mM

Preparation:

1. Measure and add BSA, pen/strep and glutamine to MEM, keep everything sterile. Culture medium can be stored at 4°C
2. On the day of experiments, pipette 50 ml of MC culture medium into 50-ml tube
3. Equilibrate in incubator (5% CO_2) for 2-3 hours prior to use. Use on same day. Do not store!

Appendix E. Mouse colony maintenance

Breeding and genotyping protocols for animals used in the thesis research are given below. These animals include the Desmin deletion (D7-des) and αB-Crystallin mutant (R120G-CryAB) mice described in chapter 6 as well as lamin A/C deficient mice and mice with a mutation in the myosin heavy chain gene (MHC Arg403). In addition, we maintained colonies with mice overexpressing Desmin and αB-Crystallin wild-type forms as additional controls. For each mice strain, one to three breeding pairs of 12-26 week old mice were maintained. Litters were weaned and genotyped at 3 weeks of age using one of the following two methods. (*i*) Buccal cells were collected for DNA extraction for Desmin, CryAB mice and MHC mice (both wild-type and mutant strains), while (*ii*) tail DNA was extracted for lamin A/C mice.

Buccal cell DNA extraction. Buccal cells were collected at time of weaning by swabbing the inner cheek ~5-10 times with a wetted cytology brush (Puritan cytology brush 2199, Harwood Products Company LP, Guilford, ME). DNA was extracted using the QuickExtract DNA extraction solution (Epicentre, Madison, WI). Cytology brushes with buccal cells were swirled in 1.7-ml Eppendorf tubes containing 100 µl of the QuickExtract solution. Subsequently, the solutions for each animal were vortexed for 20 s and transferred into a 96-well plate. The plate was then placed in a PCR machine running the following DNA extraction program.

1) 65°C for 30 min
2) 98°C for 16 min

The DNA solution was then used immediately for PCR amplification or stored at -20 °C for later analysis.

Tail DNA extraction. DNA extraction from mouse tails was performed using the Wizard Genomic DNA Purification Kit (A1120, Promega, Madison, WI). Short pieces (0.5-1 cm) of mouse tail were collected at time of weaning. Tails were digested by overnight incubation at 60 °C in 300 µl of lysis mix, consisting of 250 µl of Nuclei Lysis Solution, 60 µl of EDTA (0.5 M, pH 8.0), and 10 µl of proteinase k (20 mg/ml). Subsequently, 200

μl protein precipitation solution was added, the samples were vortexed and chilled on ice for 5 min. Following 5 min of centrifugation (13,000 rpm, 4 °C), the supernatant was transferred to a new tube containing 500 μl isopropanol and gently mixed by inversion. After incubating for 5 min at room temperature, samples were centrifuged for 5 min (13,000 rpm, room temperature) and the supernatant was decanted. The pellet was resuspended in 500 μl of 70% ethanol and centrifuged again for 5 min (13,000 rpm, room temperature). Supernatant was discarded and pellets were air-dried for 10-15 min. DNA was then rehydrated in 100 μl of DNA Rehydration solution for 1 hour at 65 °C.

Desmin wild type and D7-des mutant mice. Transgenic mice expressing either murine wild type desmin or a 7–amino acid deletion (R173 through E179) desmin mutation (D7-des) linked to desmin-related myopathy were obtained from Jeffrey Robbins (Childrens Hospital Medical Center, Division of Molecular Cardiovascular Biology, 3333 Burnett Avenue, Cincinnati, OH 45229). Transgenic mice were made and kept in the FVB/n background. Heterozygous wild type desmin or D7-des male mice were crossed with wild-type female animals to produce wild type and heterozygous offspring. Mice were genotyped by collecting buccal cell DNA and amplifying desmin DNA using polymerase chain reaction (PCR) and running the products on a 2% agarose gel. Wild-type mice have only one band of endogenous desmin, while transgenic mice have a second (smaller) band with the transgenic (wild-type or D7-des) desmin. PCR parameters are given below:

Primers: Desmin-forward: 5' cag ctt cag gaa cag cag gtc c -3'
Desmin-reverse: 5' cat caa tct cgc agg tgt agg act g -3'.

Reagents: 5 μl buccal cell DNA

15 μl PCR master mix:
- 15.1 μl ddH_2O
- 2 μl PCR buffer (10X, Sigma)
- 0.8 μl $MgCl_2$ (50 mM, GIBCO)
- 0.4 μl dNTPs (10 mM, Sigma)
- 0.1 μl sense primer (10 μM)
- 0.1 μl sense primer (10 μM)
- 0.5 μl Red Taq (1 unit/μl, Sigma)

PCR-Program step 1 - 2min 94°C

step 2 - 15 sec 94°C, 1 min 65°C, 30 sec 72°C - 30 cycles

step 3 - 10 min 72°C

step 4 – 4°C

αB-Crystallin wild type and R120G-CryAB mutant mice. Transgenic mice overexpressing either murine wild-type CryAB or the R120G mutation in cardiomyocytes were obtained from Jeffrey Robbins (see address above). Transgenic mice were made and kept in the FVB/n background. Heterozygous wild type CryAB or R120G-CryAB male mice were crossed with wild-type female animals to produce wild-type and heterozygous offspring. Mice were genotyped by extracting DNA from buccal cells and amplifying CryAB DNA using PCR with the following primers. CryAB-forward: 5' ctg gcg ttc ttc gtg ctt gcc gtg -3'; CryAB-reverse: 5' gag tct gac ctc ttc tca aca gcc -3', using the same PCR protocol as described for the Desmin mice. PCR products were run on a 2% agarose gel. Wild-type mice have only one band of endogenous CryAB, while transgenic mice have a second (smaller) band with the transgenic (wild-type or R120G) CryAB.

Lamin mice. Heterozygous (*Lmna*$^{+/-}$) mice (129SvEv background) were obtained from Colin Stewart (Cancer and Developmental Biology Lab, National Cancer Institute, P.O. Box B, Frederick, MD 21702, USA). Heterozygotes males and females were crossed to produce wild-type (*Lmna*$^{+/+}$), heterozygous (*Lmna*$^{+/-}$) and Lamin A/C null (*Lmna*$^{-/-}$) mice. Colonies were genotyped by PCR of tail DNA and by running the products on a 2% agarose gel. Wild-type mice have only one band of endogenous wild-type lamin, heterozygous mice have two bands, the larger one being the wild type and a smaller band with the genetically modified lamin, and lamin A/C deficient mice have only the smaller band. PCR parameters are given below:

Primers:	Lmna-fwd:	5' acaggtctcccaagtccccatcac -3'
	Lmna-reverse:	5' ccaggaggtaggagcgggtgact -3'
	PGKneo-rev	5' gcgctaccggtggatgtggaatgt -3'

Reagents: 2 µl tail DNA

23 µl PCR master mix: 18.05 µl ddH_2O

2.5 µl PCR buffer (10X, Sigma)

0.75 µl $MgCl_2$ (50 mM, GIBCO)

0.4 µl dNTPs (10 mM, Sigma)

0.1 µl lmna-fwd primer (10 µM)

0.1 µl lmna-rev primer (10 µM)

0.1 µl PGKneo primer (10 µM)

1.0 µl Red Taq (1 unit/µl, Sigma)

PCR-Program step 1 – 5 min 94°C

step 2 - 30 sec 94°C, 30 sec 60°C, 45 sec 72°C - 35 cycles

step 3 - 10 min 72°C

step 4 – 4 °C

MHC403 mice. Heterozygous male mice (129 SvEv background) with the MHC Arg403Gln point mutation were obtained from Jonathan Seidman (Harvard Medical School, Warren Alpert Building, Rm 524, 200 Longwood Ave, Boston, MA 02115) and were crossed with wild-type females (129 SvEv background) to produce wild type and heterozygous offspring. Mice were genotyped by PCR of buccal cell DNA using the following primers and conditions:

1) 5'-GCT GGG ACA AAG GAA TGG AGG TA -3'
2) 5'- CTG ATG GTC TGA GTG GGT AGG TGA G -3'

The annealing temperature was 65°C and the final $MgCl_2$ concentration was 1.5mM $MgCl_2$. A digestion step (Ava-I, 3-6 hours at 37°C) following the PCR allowed identification of wild-type and Arg403 mutant myosin heavy chain. PCR products were run on a 2% agarose gel.

Appendix F. Matlab programs

The following programs were written as part of the work performed for this thesis. I have tried to include the latest version of each program, written for MATLAB v6.5, but since many of these programs use proprietary MATLAB functions which may change from version to version, there is no guarantee that they will work on future version.

Image analysis programs

Multiple particle tracking program "MPTD"

```
function ud = MPTD(action, varargin)
% Multiple Particle Tracking Program with GUI
% requires the MATLAB files: gaussfit.m parabolicfit.m forthorderfit.m
% Jan Lammerding
% 02/07/2001
% added          1) threshold for cross-correlation,
%                       2) invert signal for brighfield for crosscorrelation
%                       3) close result windows when exiting program
% 6/16/2001
% fixed findnextfile function
% 7/23/01
% started to converted for MATLAB 6.1.0 and reading avi-files from cd
% 7/27/01
% completed avi-conversion
% corrected fminsearch function
% corrected parabolicfit.m
% 7/28/01
% implemented new cross-correlation algorithm
% 7/29/01
% allowed standard menu in result windows
% allowed user choice between CD rom and HD access for athena stations
% 11/13/01
% inactivated system choice (use PC as default)
% added directory memory
% disabled kernel update after every frame since this lead to kernel shift.
% fixed matched kernel algorithm to guarantee that area is within frame
% 03/24/02
% fixed searchArea condition to make sure that search area is larger than kernel area
% introduced criteria for valid measurement: x-correlation maximum>0.5, otherwise center=[-1 -1]
% don't do x-correlation after bead detachment
% 5/29/02
% modified key funcitons: delele- delete bead, return - start tracking
% 6/7/02
% corrected SearchArea boundary calculation to avoid "index exceeds matrix"-error.
% included option to change minimum x-correlation value by hitting x or p key
% renamed MPTD for Multiple Particle Tracking + Data Reader
% reads magnetic field data from movie (top left pixel) and stores it along with the tracking results
% 9/9/03
% update HitTest property for region markers to allow selecting overlapping
% objects
```

```
if nargin<1,
   action='InitializeMPTD';
   if nargout>0  % If user wants the UserData, give it to them.
      ud = feval(action,varargin{:});
      return
   end
end

feval(action,varargin{:})
return


%-----------------------------------------------
%
% subfunction: initialize multiple particle tracking (MPTD)
%
function udout = InitializeMPTD()
% If MPTD is already running, bring it to the foreground
h = findobj(allchild(0), 'tag', 'MPTD Program');
if ~isempty(h)
   figure(h(1))
   if nargout>0
      udout = get(h(1), 'UserData');
   end
   return
end
% Initialize MPTD window:
% use default path:
ud.h.CDpath=pwd;       % directory with avi-files
ud.h.Savepath=pwd;     % directory to store results

MPTFig = figure('Color',[0.8 0.8 0.8], ...
   'Colormap', gray(256), ...
   'MenuBar','none', ...
   'BusyAction','Queue','Interruptible','off',...
   'Name','Multiple Particle Tracking Program + Data Reader', ...
   'Tag','MPTD Program',...
   'Visible','off','Resize','off','HandleVisibility', 'on',...
   'NumberTitle','off', 'IntegerHandle', 'off', ...
   'Position',[350 230 610 460], ...
   'WindowButtonMotionFcn','MPTD("NormalMotionFcn");',...
   'KeyPressFcn','MPTD("KeyPress");',...
   'CloseRequestFcn','MPTD("CloseMPT");');
ud.FigureHandle = MPTFig;
% define Standard Values
ud.Beads=0;
ud.Method=[];
ud.Filter=[];
ud.Field=[];
ud.KernelWidth=[];
ud.KernelHeight=[];
ud.SearchWidth=[];
ud.SearchHeight=[];
ud.KernelPosition=[];
ud.XcorrThr=0.5;
ud.Pathname=pwd;
```

```
ud.Filename='MyFile';
ud.Frames=0;
ud.Results=[];
ud.h.oldpath=pwd;    %store original pathname
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% handles for bead markers
ud.h.Rect=[];
ud.h.Label=[];
% define standard settings
Std.Interruptible = 'off';
Std.BusyAction = 'queue';

% Defaults for image axes
Ax = Std;
Ax.Units = 'Pixels';
Ax.Parent = MPTFig;
Ax.XTick = [];
Ax.YTick = [];
Ax.Box='on';

Img = Std;
Img.CData = [];
%Img.CDataMapping = 'Scaled';

Ctl = Std;
Ctl.Units = 'Pixels';
Ctl.Parent = MPTFig;

Btn=Ctl;
Btn.Parent = MPTFig;
Btn.Style = 'pushbutton';
Btn.Enable = 'off';

Frame = Ctl;
Frame.Style = 'Frame';

Edit = Ctl;
Edit.Style = 'edit';
Edit.HorizontalAlignment = 'right';
Edit.BackgroundColor = 'white';
Edit.ForegroundColor = 'black';

Menu = Ctl;
Menu.Style = 'Popupmenu';

Text = Ctl;
Text.Style = 'text';
Text.HorizontalAlignment = 'left';


% Frames
bgd=[0.7 0.7 0.7];
Frame1 = uicontrol(Frame, ...
   'Position',[455 125 145 320]);
Frame2 = uicontrol(Frame, ...
   'Position',[155 10 445 100]);
```

```
Frame3 = uicontrol(Frame, ...
   'BackgroundColor',bgd,...
   'Position',[ 415 63 175 40 ]);
Frame4 = uicontrol(Frame, ...
   'BackgroundColor',bgd,...
   'Position',[ 415 17 175 40 ]);

% Images
StartImg=200*ones(480,640);   %start image: gray screen
Kernel=ones(20,20);                                    % initial kernel display: black

ud.h.ImageAxes = axes(Ax, ...
   'Position', [35 145 400 300]);
ud.h.ImageImage = image(StartImg, Img,...
   'Parent',ud.h.ImageAxes, ...
   'XData',[1 size(StartImg,2)],'YData',[1 size(StartImg,1)],...
   'ButtonDownFcn','MPTD("PickBead")'); %, ...
set(ud.h.ImageImage,'XData',[1 640],'YData',[1 480]);
set(ud.h.ImageAxes,'DataAspectRatio',[1 1 1],'TickDir','in')
ud.h.TrackBox = line('XData', [], ...
   'YData', [], ...
   'Parent',ud.h.ImageAxes,...
   'Clipping', 'on', ...
   'Color', 'r', ...
   'LineStyle', '-', ...
   'EraseMode', 'xor',...
   'Visible','off');
udKernelFrame = uicontrol(Frame, ...
   'BackgroundColor',bgd,...
   'Position',[54 6 53 16]);
udKernelTitel = uicontrol(Text, ...
   'BackgroundColor',bgd,...
   'Position',[56 8 49 12], ...
   'HorizontalAlignment','center', ...
   'String','Kernel');
ud.h.KernelAxes= axes(Ax, ...
   'Position', [38 24 85 85]);
ud.h.KernelImage= image(Kernel, Img, ...
   'Parent', ud.h.KernelAxes);
set(ud.h.KernelAxes,'PlotBoxAspectRatio',[1 1 1], ...
   'DataAspectRatio',[1 1 1],'XTick',[],'YTick',[],...
   'XLimMode','auto','YLimMode','auto');
ud.h.Cross= line('XData', [18 22 20 20 20], ...
   'YData', [20 20 20 18 22], ...
   'Clipping', 'on', ...
   'Color', 'r', ...
   'Parent',ud.h.KernelAxes,...
   'LineStyle', '-', ...
   'EraseMode', 'normal',...
   'Visible','off');

% Pushbuttons
ud.h.LoadData = uicontrol(Btn, ...
   'Position',[465 400 125 35], ...
   'Enable','on',...
   'String','Load AVI',...
```

```
   'Callback','MPTD("LoadFrame")');
ud.h.LoadParameters = uicontrol(Btn, ...
   'Position',[465 347 125 35], ...
   'String','Load Parameters', ...
   'Enable','on',...
   'Callback','MPTD("LoadParameters")');
ud.h.SaveParameters = uicontrol(Btn, ...
   'Position',[465 294 125 35], ...
   'String','Save Parameters',...
   'Callback','MPTD("SaveParameters")');
ud.h.StartTracking = uicontrol(Btn, ...
   'Position',[465 241 125 35], ...
   'String','Start Tracking',...
   'Callback','MPTD("StartTracking")');
udhInfo = uicontrol(Btn, ...
   'Position',[465 188 125 35], ...
   'Enable','on',...
   'String','Info',...
   'Callback','MPTD("DisplayInfo")');
udhQuit = uicontrol(Btn, ...
   'Position',[465 135 125 35], ...
   'String','Quit', ...
   'Enable','on',...
   'Callback','MPTD("CloseMPT")');
ud.h.Prev = uicontrol(Btn, ...
   'Position',[165 80 115 22], ...
   'String','Prev',...
   'Callback','MPTD("PreviousBead")');
ud.h.Next = uicontrol(Btn, ...
   'Position',[165 18 115 22], ...
   'String','Next',...
   'Callback','MPTD("NextBead")');
h1 = uicontrol(Text, ...
   'FontWeight','bold', ...
   'Position',[165 47 40 26], ...
   'String',['Curr.';'Bead ']);
ud.h.CurrentBead = uicontrol(Edit, ...
   'Position',[205 47 25 26], ...
   'String','1',...
   'Callback','MPTD("ChangeBead")');
ud.h.Delete = uicontrol(Btn, ...
   'Position',[240 47 40 26], ...
   'String','Delete',...
   'Callback','MPTD("DeleteBead")');
ud.h.Method = uicontrol(Menu, ...
   'Position',[290 77 115 25], ...
   'String',{'Center of Mass','Gauss','Parabolic'}, ...
   'Value',1,...
   'Callback','MPTD("ChangeParameters")');
ud.h.Filter = uicontrol(Menu, ...
   'Position',[290 46 115 25], ...
   'String',{'No Filter','Threshold Filter'}, ...
   'Value',2,...
   'Callback','MPTD("ChangeParameters")');
ud.h.Field = uicontrol(Menu, ...
   'Position',[290 15 115 25], ...
```

```
   'String',{'Brightfield','Darkfield'}, ...
   'Value',1,...
   'Callback','MPTD("ChangeParameters")');
h1 = uicontrol(Text, ...
   'BackgroundColor',bgd,...
   'Position',[420 72 63 20], ...
   'String','Kernel Size');
h1 = uicontrol(Text, ...
   'BackgroundColor',bgd,...
   'Position',[490 81 33 20], ...
   'String','Width');
h1 = uicontrol(Text, ...
   'BackgroundColor',bgd,...
   'Position',[545 81 33 20], ...
   'String','Height');
ud.h.KernelWidth = uicontrol(Edit, ...
   'Position',[490 67 33 20], ...
   'String','20',...
   'Callback','MPTD("ChangeParameters")');
ud.h.KernelHeight = uicontrol(Edit, ...
   'Position',[545 67 33 20], ...
   'String','20',...
   'Callback','MPTD("ChangeParameters")');
h1 = uicontrol(Text, ...
   'Position',[420 26 63 20], ...
   'BackgroundColor',bgd,...
   'String','Search Area');
h1 = uicontrol(Text, ...
   'Position',[545 35 33 20], ...
   'BackgroundColor',bgd,...
   'String','Height');
h1 = uicontrol(Text, ...
   'Position',[490 35 33 20], ...
   'BackgroundColor',bgd,...
   'String','Width');
ud.h.SearchWidth = uicontrol(Edit, ...
   'Position',[490 21 33 20], ...
   'String','3',...
   'Callback','MPTD("ChangeParameters")');
ud.h.SearchHeight = uicontrol(Edit, ...
   'Position',[545 21 33 20], ...
   'String','1',...
   'Callback','MPTD("ChangeParameters")');
set(MPTFig, 'Userdata', ud);
%set(MPTFig, 'visible','on','HandleVisibility','callback');
set(MPTFig, 'visible','on');

if nargout>0                    % return userdata if requested
   udout = ud;
end
return

%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Jan's subfunctions
%%  Sub-Function - NormalMotionFcn
%%
```

```
function NormalMotionFcn
% Set the cursor to a Cross-Hair when above the Original image, and back
% to an arrow when not.
% This is the normal motion function for the window when we are not in
% a MyGetline selection state.
DemoFig = gcbf;
ud = get(DemoFig, 'Userdata');
pos = get(ud.h.ImageAxes, 'Position');
pt = get(DemoFig, 'CurrentPoint');
x = pt(1,1);
y = pt(1,2);
if (x>pos(1) & x<pos(1)+pos(3) & y>pos(2) & y<pos(2)+pos(4))
   set(DemoFig,'Pointer','crosshair');
   % follow mouse with rectangle
   pt2 = get(ud.h.ImageAxes, 'CurrentPoint');
   KX=str2num(get(ud.h.KernelWidth,'String'));
   KY=str2num(get(ud.h.KernelHeight,'String'));
   KernelX=pt2(1,1);
   KernelY=pt2(1,2);
   X=[KernelX-KX/2 KernelX+KX/2 KernelX+KX/2 KernelX-KX/2 KernelX-KX/2];
   Y=[KernelY-KY/2 KernelY-KY/2 KernelY+KY/2 KernelY+KY/2 KernelY-KY/2];
   set([ud.h.TrackBox], 'XData', X, 'YData', Y);
   set([ud.h.TrackBox], 'Visible','on');
else
   set(DemoFig, 'Pointer', 'arrow');
   % hide box
   set([ud.h.TrackBox],'Visible','off')
end


%% Sub-Function - CloseMPT
%%
function CloseMPT
% Close request function for the MPTD program
% User-defined close request function
% to display a question dialog box
selection = questdlg('Do you really want to quit ?',...
   'Quit MPTD Request',...
   'Yes','No','Yes');
switch selection,
case 'Yes',
   delete(gcf)
   for i=1:5
      h = findobj(allchild(0), 'tag', ['MPTD Results' num2str(i)]);
      if ~isempty(h)                    % close Result windows
         close(h)
      end
   end
case 'No'
   return
end


%%  Sub-Function - PickBead
%%
function PickBead
```

```
DemoFig = gcbf;
ud = get(DemoFig, 'UserData');

pt = round(get(gca, 'CurrentPoint'));
N=str2num(get(ud.h.CurrentBead,'String'));
Beads=ud.Beads;
KX=str2num(get(ud.h.KernelWidth,'String'));
KY=str2num(get(ud.h.KernelHeight,'String'));

KernelX=pt(1,1);
KernelY=pt(1,2);
RectX=[KernelX-KX/2 KernelX+KX/2 KernelX+KX/2 KernelX-KX/2 KernelX-KX/2];
RectY=[KernelY-KY/2 KernelY-KY/2 KernelY+KY/2 KernelY+KY/2 KernelY-KY/2];
% check if bead exists
if N>ud.Beads                          % new bead
   ud.Beads=N;
else                                                    % erase old bead and overwrite properties
   delete(ud.h.Label(N));
   delete(ud.h.Rect(N));
end
% Initialize the rectangles to mark beads
ud.h.Rect(N) = line('XData', RectX, ...
   'YData', RectY, ...
   'Parent', ud.h.ImageAxes, ...
   'Clipping', 'on', ...
   'Color', 'b', ...
   'LineStyle', '-', ...
   'HitTest', 'off', ...
   'EraseMode', 'normal');                    % change this back to xor and use 'k' as color
ud.h.Label(N) = text(RectX(3)+2,RectY(3),num2str(N), ...
   'Parent',ud.h.ImageAxes,'EraseMode','normal','color','b','HitTest','off');
ud.Method(N)=get(ud.h.Method,'Value');
ud.Filter(N)=get(ud.h.Filter,'Value');
ud.Field(N)=get(ud.h.Field,'Value');
ud.KernelWidth(N)=KX;
ud.KernelHeight(N)=KY;
ud.SearchWidth(N)=str2num(get(ud.h.SearchWidth,'String'));
ud.SearchHeight(N)=str2num(get(ud.h.SearchHeight,'String'));
ud.KernelPosition(1:2,N)=[KernelX; KernelY];
% update userdata:
set(ud.h.TrackBox,'XData',get(ud.h.TrackBox,'XData')-1000);          % to avoid overlapping !
set(ud.h.CurrentBead,'String',num2str(N+1));
set([ud.h.Prev ud.h.SaveParameters ud.h.StartTracking],'Enable','on')
set([ud.h.Next ud.h.Delete],'Enable','off');
set(DemoFig, 'UserData', ud);
UpdateKernel


%%
%% Sub-Function LoadFrame
function LoadFrame
DemoFig = gcbf;
ud = get(DemoFig, 'UserData');
cd(ud.h.CDpath)    %set path for avi-directory
[filename, pathname]=uigetfile('*.avi','Select avi-file');
if filename
```

```
  set(DemoFig,'Pointer','watch')
  file=fullfile(pathname,filename);
  avi=aviread(file,1);
  infoavi=aviinfo(file);
  % update Userdata
  img=double(avi.cdata);
  ud.h.CDpath=pathname;  % store this path as default for avi-directory
  ud.Pathname=pathname;
  ud.Filename=filename;
  ud.Frames=infoavi.NumFrames;    %number of frames
  set(ud.h.ImageImage, 'Cdata', img,...
    'XData',[1 infoavi.Width],'YData',[1 infoavi.Height]);
  set(ud.h.ImageAxes,'XLim',[1 infoavi.Width],'YLim',[1 infoavi.Height]);
  set(DemoFig,'UserData',ud);
  UpdateNumbers
  UpdatePopup
  UpdateKernel
  NormalMotionFcn
  %display msgbox with avi-file information
  information=char(' FILE INFORMATION','----------------',...
    ['file name:       ' infoavi.Filename], ...
    ['last modified:   ' infoavi.FileModDate],...
    ['file size:       ' num2str(infoavi.FileSize)], ...
    ['number of frames: ' num2str(infoavi.NumFrames)]);
  %msgbox(information)
  cd(ud.h.oldpath)  % reset old path
end


%%
%% Sub-function SaveParameters
function SaveParameters
DemoFig = gcbf;
ud = get(DemoFig, 'UserData');
%oldDir=cd;       % remember old directory
cd(ud.h.Savepath)   % switch to directory for saving results
[path, name, ext, ver]=fileparts(ud.Filename);
initfile=[name '_parameters' '.mat'];
[filename, pathname]=uiputfile(initfile,'Save Parameters');
%reset directory to old directory:
cd(ud.h.oldpath);
if filename
  file=fullfile(pathname,filename);
  ud.h.Savepath=pathname;
  set(DemoFig,'UserData',ud);
  % prepare filestructure
  data=ud; data.FigureHandle=[]; data.h=[]; data.Results=[];
  save(file,'data');
end


%%
%% Sub-function LoadParameters
function LoadParameters
DemoFig = gcbf;
ud = get(DemoFig, 'UserData');
```

```
%oldDir=cd;
cd(ud.h.Savepath)
[filename, pathname]=uigetfile('*.mat','Load Parameters');
cd(ud.h.oldpath);
if filename
   set(DemoFig,'Pointer','watch')
   data=[];
   file=fullfile(pathname,filename);
   load(file)
   % erase old bead markers
   for i=1:ud.Beads
      delete(ud.h.Label(i));
      delete(ud.h.Rect(i));
   end
   ud.h.Rect=[];
   ud.h.Label=[];
   ud.h.Savepath=pathname;
   % update Userdata
   data.h=ud.h; data.FigureHandle=ud.FigureHandle;          % keep Figurehande and guicontrols
   ud=data;
   set(DemoFig,'UserData',ud);
   set(ud.h.CurrentBead,'String','1')
   % update enable buttons
   set([ud.h.Next ud.h.Delete ud.h.StartTracking ud.h.SaveParameters],'Enable','on');
   set(ud.h.Prev,'Enable','off')
   if ud.Beads<=1
      set(ud.h.Next,'Enable','off')
   end
   % update kernels
   for i=1:ud.Beads
      KernelX=ud.KernelPosition(1,i);
      KernelY=ud.KernelPosition(2,i);
      KX=ud.KernelWidth(1,i);
      KY=ud.KernelHeight(1,i);
      RectX=[KernelX-KX/2 KernelX+KX/2 KernelX+KX/2 KernelX-KX/2 KernelX-KX/2];
      RectY=[KernelY-KY/2 KernelY-KY/2 KernelY+KY/2 KernelY+KY/2 KernelY-KY/2];
      ud.h.Rect(i) = line('XData', RectX, ...
         'YData', RectY, ...
         'Clipping', 'on', ...
         'Parent', ud.h.ImageAxes, ...
         'Color', 'b', ...
         'LineStyle', '-', ...
         'EraseMode', 'normal');              % this used to be xor and color 'k'
      ud.h.Label(i) = text(RectX(3)+2,RectY(3),num2str(i), ...
         'Parent', ud.h.ImageAxes, 'EraseMode','normal');
      set(ud.h.Label(i),'Color','b')
   end
   set(DemoFig,'UserData',ud);
   UpdateNumbers
   UpdatePopup
   UpdateKernel
   NormalMotionFcn
end


%%
```

```
%% Sub-Function DisplayInfo
function DisplayInfo
str={'Getting started' ...
    ['Load the avi-file you want to perform the particle tracking on. The 1st '
    'frame will be displayed, along with some general info on the file.    '
    'Set the parameters (Centroid determination method,                    '
    'filter, etc.) to the appropiate settings and then select the beads you '
    'want to track by clicking on them with the left mouse button. You can  '
    'modify the parameters for each bead by using the <Prev> or <Next> button'
    'to go to the bead and then adjust the parameters. The kernel image will '
    'be updated immediately.                                               '
    'When done, save the current parameters (Kernel positions and settings) '
    'and press <Start Tracking> to begin tracking. You will be prompted to  '
    'enter the file to which the results are save. If you hit <cancel>, the '
    'results will not be saved.                                        '];...
    'Parameters' ...
    ['The following parameters can be selected for each bead               '
    '                                                                      '
    'Center of Mass:  The centroid position is calculated based on the center'
    '                 of mass (the intensity). This is the fastest and most '
    '                 robust algorithm, but not the most accurate on.      '
    'Gauss:           Fits 2-dimensional Gauss-distribution to kernel      '
    '                 intensity to determine the center location based on  '
    '                 iterative least-square fit.                          '
    'Parabolic:       Fits 2nd order polynomial to kernel intensity map to '
    '                 determine center location based on least square fit. '
    '                                                                      '
    'Kernel size:     The kernel size in pixels. Kernels are centered around '
    '                 the point that was selected with the mouse.          '
    'Search area:     Area for which the cross-correlation is performed to '
    '                 identify the matching kernel in the following frame. '
    '                 The search area is centered around the kernel.       '
    '                                                    '];...
    'File format' ...
    ['Video file:     Video should be grayscale images of type ".avi", with '
    '                 intensity values ranging from 0 (black) to 255 (white).'
    'Parameter file:  File thatcontains the kernel positions and settings. '
    '                 This is a MATLAB ".mat" file.                        '
    'Result file:     Results are saved as comma-separated-value (*.csv)   '
    '                 files. Each bead is represented by a pair of columns. '
    '                 The first column contains the x-position for the first '
    '                 bead, the second column the y-position. Each row     '
    '                 represents the position for one frame.             ']};
helpwin(str,'Getting started','Multiple Particle Tracking Program')


%%
%% Sub-function FindCenter
function CenterPosition=FindCenter(KernelImg,method,field,filter)
% N: Bead Number
% KernelImg: bitmap of kernel
% method: 1: center of mass, 2: gauss, 3: parabolic
% field: 1: brightfield 2: darkfield
% filter: 1: no filter, 2: threshold filter
% center is being expressed relative to kernel (topleft=1,1)
% requires the files gaussfit.m, parabolicfit.m, and forthorderfit.m
```

```
%%
if field==1                                                    % invert image
for bright field images
  KernelData=max(max(KernelImg))-KernelImg;
else
  KernelData=KernelImg;
end
if filter==2                                                   % Threshold filter
  T=mean2(KernelData);              % use mean as Threshold
  KernelData(KernelData<T)=0;       % set values below threshold to zero
end
% set iteration tol (1e-6) and max iteration (1000):
options=optimset('Display','off','TolX',1e-4,'TolFun',1e-6,'MaxIter',1000);
% use estimate for center:
[h w]=size(KernelData);                                % determine size of kernel
switch method
case 1
  % CENTER OF MASS FIT
  [x,y]=meshgrid(1:w,1:h);
  xc=sum(sum(x.*KernelData))/(sum(sum(KernelData)+eps));       % compute center of intensity
  yc=sum(sum(y.*KernelData))/(sum(sum(KernelData)+eps));
case 2
  % GAUSSIAN FIT
  C0=[w/2 h/2 1 max(max(KernelData))/1.995];          % start values for iteration
  center=fminsearch('gaussfit',C0,options,KernelData);
  xc=center(1);
  yc=center(2);
case 3
  % PARABOLIC FIT
  C0=[w/2 h/2 1 max(max(KernelData))];
  center=fminsearch('parabolicfit',C0,options,KernelData);
  xc=center(1);
  yc=center(2);
otherwise                                                      % invalid method selection
  disp('invalid method selection')
  xc=-1;
  yc=-1;
end
if (xc<1 | xc>w | yc<1 | yc>h)          % return -1,-1 in case of invalid results
  xc=-1;
  yc=-1;
end
CenterPosition=[xc; yc];


%%
%% Sub-function KeyPress
function KeyPress
DemoFig = gcbf;
ud = get(DemoFig, 'UserData');
key = real(get(DemoFig, 'CurrentCharacter'));
if isempty(key), % switch statement was triggering error in Unix, the
  return;     % "Shift" key was causing the variable key to be []
end
switch key
case 8  % backspace keys
```

```
   PreviousBead              % same as prev-bead
case {13, 3}   % enter and return keys
   %NextBead
   StartTracking
case 27                      % escape key
   CloseMPT
case 127       % delete key
   DeleteBead
case {120, 88}  % x or X
   EditParameter
case {112, 80}  % p or P
   EditParameter
end


%%
%% Sub-Function PreviousBead
function PreviousBead
DemoFig = gcbf;
ud = get(DemoFig, 'UserData');
N=str2num(get(ud.h.CurrentBead,'string'))-1;
if N>=1              % make sure that current bead>=1
   set(ud.h.CurrentBead,'String',num2str(N));
   set([ud.h.Next ud.h.Delete],'Enable','on')
   UpdatePopup
   UpdateNumbers
   UpdateKernel
end
if N==1              % disable prev-button once bead #1 is reached
   set(ud.h.Prev,'Enable','off');
end


%%
%% Sub-Function NextBead
function NextBead
DemoFig = gcbf;
ud = get(DemoFig, 'UserData');
N=str2num(get(ud.h.CurrentBead,'string'))+1;          %increase beadnumber by 1
if N<=ud.Beads+1          % make sure that current bead<total beads
   set(ud.h.CurrentBead,'String',num2str(N));
   set(ud.h.Prev,'Enable','on');
   UpdatePopup
   UpdateNumbers
   UpdateKernel
end
if (N==ud.Beads+1)                  % disable next-button once new bead (last+1) is reached
   set([ud.h.Next ud.h.Delete],'Enable','off');
end


%%
%% Sub-function UpdateKernel
function UpdateKernel
DemoFig = gcbf;
ud = get(DemoFig, 'UserData');
```

```
N=str2num(get(ud.h.CurrentBead,'string'));
if N<=ud.Beads                          % only display kernel if bead has been picked
   KernelX=ud.KernelPosition(1,N);
   KernelY=ud.KernelPosition(2,N);
   KX=ud.KernelWidth(N);
   KY=ud.KernelHeight(N);
   FrameImg=get(ud.h.ImageImage,'Cdata');
   [maxY,maxX]=size(FrameImg);          % determine max values from frame size
   LeftX=max([round(KernelX-KX/2) 1]);
   RightX=min([round(KernelX+KX/2) maxX]);
   TopY=max([1 round(KernelY-KY/2)]);
   BottomY=min([round(KernelY+KY/2) maxY]);
   KernelImg=FrameImg(TopY:BottomY,LeftX:RightX);
else
   KX=str2num(get(ud.h.KernelWidth,'string'));
   KY=str2num(get(ud.h.KernelHeight,'string'));
   KernelImg=ones(KY,KX);
   set(ud.h.Cross,'Visible','off');
 end
 [h,w]=size(KernelImg);
 set(ud.h.KernelImage, 'Cdata', KernelImg,'XData',[1 w],'YData',[1 h]);
 % mark centroid position
 if N<=ud.Beads
   C=FindCenter(KernelImg,ud.Method(N),ud.Field(N),ud.Filter(N));
   if C~=[-1; -1]  % in case we found a center
     s=2;                                % size of centroid marker
     set(ud.h.Cross,'XData', [C(1)-s C(1)+s C(1) C(1) C(1)], ...
       'YData', [C(2) C(2) C(2) C(2)-s C(2)+s], ...
       'Visible','on');
   else
     set(ud.h.Cross,'Visible','off');
   end
 end


%%
%% Sub-function UpdatePopup
function UpdatePopup
DemoFig = gcbf;
ud = get(DemoFig, 'UserData');
N=str2num(get(ud.h.CurrentBead,'string'));
if N<=ud.Beads                          % only update Popups if bead has been picked
   set(ud.h.Method,'Value',ud.Method(N));
   set(ud.h.Filter,'Value',ud.Filter(N));
   set(ud.h.Field,'Value',ud.Field(N));
end


%%
%% Sub-function UpdateNumbers
function UpdateNumbers
DemoFig = gcbf;
ud = get(DemoFig, 'UserData');
N=str2num(get(ud.h.CurrentBead,'string'));
if N<=ud.Beads                          % only update Numbers if bead has been picked
   set(ud.h.KernelWidth,'String',num2str(ud.KernelWidth(N)));
```

```
   set(ud.h.KernelHeight,'String',num2str(ud.KernelHeight(N)));
   set(ud.h.SearchWidth,'String',num2str(ud.SearchWidth(N)));
   set(ud.h.SearchHeight,'String',num2str(ud.SearchHeight(N)));
end


%%
%% Sub-function ChangeBead
function ChangeBead
DemoFig = gcbf;
ud = get(DemoFig, 'UserData');
N=str2num(get(ud.h.CurrentBead,'string'));
N=median([1 N ud.Beads+1]);                    % make sure that bead# is in allowed range
set(ud.h.CurrentBead,'String',num2str(N));
set([ud.h.Prev ud.h.Next ud.h.Delete],'Enable','on');
if N==1;
   set(ud.h.Prev,'Enable','off');
end
if N==ud.Beads+1;
   set([ud.h.Next ud.h.Delete],'Enable','off');
end
UpdatePopup
UpdateNumbers
UpdateKernel                              % update Kernel


%%
%% Sub-function ChangeParameters
function ChangeParameters
DemoFig = gcbf;
ud = get(DemoFig, 'UserData');
N=str2num(get(ud.h.CurrentBead,'string'));
if N<=ud.Beads
   ud.Method(N)=get(ud.h.Method,'Value');
   ud.Filter(N)=get(ud.h.Filter,'Value');
   ud.Field(N)=get(ud.h.Field,'Value');
   ud.KernelWidth(N)=str2num(get(ud.h.KernelWidth,'String'));
   ud.KernelHeight(N)=str2num(get(ud.h.KernelHeight,'String'));
   ud.SearchWidth(N)=str2num(get(ud.h.SearchWidth,'String'));
   ud.SearchHeight(N)=str2num(get(ud.h.SearchHeight,'String'));
   % update kernelmarker rectangle
   KernelX=ud.KernelPosition(1,N);
   KernelY=ud.KernelPosition(2,N);
   KX=ud.KernelWidth(N);
   KY=ud.KernelHeight(N);
   RectX=[KernelX-KX/2 KernelX+KX/2 KernelX+KX/2 KernelX-KX/2 KernelX-KX/2];
   RectY=[KernelY-KY/2 KernelY-KY/2 KernelY+KY/2 KernelY+KY/2 KernelY-KY/2];
   set(ud.h.Rect(N),'XData',RectX,'YData',RectY);
   set(ud.h.Label(N),'Position',[RectX(3)+2, RectY(3)]);
   % update label positions
   set(DemoFig,'UserData',ud)
   UpdateKernel
end


%%
```

```
%% Sub-function DeleteBead
function DeleteBead
DemoFig = gcbf;
ud = get(DemoFig, 'UserData');
N=str2num(get(ud.h.CurrentBead,'string'));
if N<=ud.Beads
   ud.Beads=ud.Beads-1;
   ud.Method=[ud.Method(1:N-1) ud.Method(N+1:end)];
   ud.Filter=[ud.Filter(1:N-1) ud.Filter(N+1:end)];
   ud.Field=[ud.Field(1:N-1) ud.Field(N+1:end)];
   ud.KernelWidth=[ud.KernelWidth(1:N-1) ud.KernelWidth(N+1:end)];
   ud.KernelHeight=[ud.KernelHeight(1:N-1) ud.KernelHeight(N+1:end)];
   ud.SearchWidth=[ud.SearchWidth(1:N-1) ud.SearchWidth(N+1:end)];
   ud.SearchHeight=[ud.SearchHeight(1:N-1) ud.SearchHeight(N+1:end)];
   ud.KernelPosition=[ud.KernelPosition(:,1:N-1) ud.KernelPosition(:,N+1:end)];
   delete(ud.h.Rect(N));
   delete(ud.h.Label(N));
   ud.h.Rect=[ud.h.Rect(1:N-1) ud.h.Rect(N+1:end)];
   ud.h.Label=[ud.h.Label(1:N-1) ud.h.Label(N+1:end)];
   set(DemoFig,'UserData',ud)
   if N>ud.Beads;            %in case we deleted the most recent bead
      set(ud.h.CurrentBead,'String',num2str(max([1 ud.Beads])));
   end
   if ud.Beads==0;
      set([ud.h.Delete ud.h.Next],'Enable','off')
   end
   if ud.Beads==1;
      set(ud.h.Prev,'Enable','off')
   end
   % update Labels
   for i=N:ud.Beads
      set(ud.h.Label(i),'String',num2str(i));
   end
   UpdateKernel
   UpdateNumbers
   UpdatePopup
end


%%
%% ------------------------------
%% Sub-Function StartTracking
function StartTracking
DemoFig = gcbf;
ud = get(DemoFig, 'UserData');
% Select File to save Results to
cd(ud.h.Savepath);
[path, name, ext, ver]=fileparts(ud.Filename);
initfile=[name '_MPTresults' '.csv'];
[filename, pathname]=uiputfile(initfile,'Save Results to');
%reset directory to old directory:
cd(ud.h.oldpath);
set(DemoFig,'Pointer','Watch')       % set pointer to hour-glass
if ud.Frames>1
   ud.Results=[];  % clear old results
   set(DemoFig,'UserData',ud)
```

```
  ParticleTracking                    % perform particleTracking
  % get results from UserData
  if filename                                % save results if user selected filename
    ud = get(DemoFig, 'UserData');
    ud.h.Savepath=pathname;
    set(DemoFig,'UserData',ud);
    [pathstr, fname, ext, vers]=fileparts(filename);
    if ~length(ext)
       filename=[filename '.csv'];
    end
    file=fullfile(pathname,filename);
    csvwrite(file,ud.Results);                % write results to csv-file
  end
  DisplayResults
else
  disp('Warning: need more than one Frame')
end
NormalMotionFcn                     % reset pointer


%%
%% Sub-Function ParticleTracking
function ParticleTracking
DemoFig = gcbf;
ud = get(DemoFig, 'UserData');
%answer=(inputdlg('Enter how many frames to process:', 'Perform Particle
Tracking',1,{num2str(ud.Frames)}));
%NFrames=str2num(answer{1});          % convert to number
%NFrames=median([0 NFrames ud.Frames]);  % make sure that 0<NFrames<max # of frames
NFrames=ud.Frames;  %process all frames
filename=fullfile(ud.Pathname,ud.Filename);
Nbead=ud.Beads;                                          % number of beads
kernels=[];                 % structure to store kernel images
LeftX=[];
TopY=[];
FrameImg=get(ud.h.ImageImage,'Cdata');
[maxY,maxX]=size(FrameImg);              % determine max values from frame size
for N=1:Nbead
  % store kernel data
  KernelX=ud.KernelPosition(1,N);
  KernelY=ud.KernelPosition(2,N);
  KX=ud.KernelWidth(N);
  KY=ud.KernelHeight(N);
  LeftX(N)=max([round(KernelX-KX/2) 1]);
  RightX=min([round(KernelX+KX/2) maxX]);
  TopY(N)=max([1 round(KernelY-KY/2)]);
  BottomY=min([round(KernelY+KY/2) maxY]);
  kernels{N}=FrameImg(TopY(N):BottomY,LeftX(N):RightX);
end
% store kernels for later comparison
ud.h.firstKernels=kernels;
% INITIALIZE RESULT VARIABLES
center=zeros(NFrames,2*Nbead);
avidata=zeros(NFrames,1); avidata(1,1)=FrameImg(1,1);
ud.h.CrosscorrelationResults=zeros(NFrames-1,Nbead);
for N=1:Nbead
```

```
    % COMPUTE CENTER OF BEAD
    kernel=kernels{N};
    offset=[LeftX(N)-1; TopY(N)-1];
    KernelCenter=FindCenter(kernel,ud.Method(N),ud.Field(N),ud.Filter(N))+offset;
    center(1,2*N-1:2*N)=KernelCenter.';                    % store first position
    % calculate starting point for cross correlation
    StartX(N)=LeftX(N)-round(ud.SearchWidth(N)/2);              % starting point x
    StartY(N)=TopY(N)-round(ud.SearchHeight(N)/2);                    % starting point y
end
%display window indicating progress
waitb=waitbar(1/NFrames,' Particle tracking in progress, please wait...');
tic         % start measuring computation time
% PERFORM PARTICLE TRACKING FOR IMAGES 2-NFrames
for n=2:NFrames
    % open next image
    avi=aviread(filename,n);
    currentFrame=avi.cdata;
    avidata(n,1)=currentFrame(1,1);
    for N=1:Nbead
        if center(n-1,2*N-1)==-1
            newCenter=[-1; -1];
        else
            kernel=double(kernels{N});
            if ud.Field(N)==1              % in case of Brightfield
                kernel=255-kernel;                  % invert kernel intensity
            end
            if ud.Filter(N)==2                                                    % Threshold
filter
                kernel(kernel<mean2(kernel))=0;             % set values below threshold (=mean) to zero
            end
            % CROSS-CORRELATION
            validCenterX=round(median([1+ud.SearchWidth(N)+ud.KernelWidth(N)/2 center(n-1,2*N-1)
maxX-ud.SearchWidth(N)-ud.KernelWidth(N)/2-1]));
            validCenterY=round(median([1+ud.SearchHeight(N)+ud.KernelHeight(N)/2 center(n-1,2*N)
maxY-ud.SearchHeight(N)-ud.KernelHeight(N)/2-1]));
            SearchAreaLX=round(validCenterX-ud.SearchWidth(N)-ud.KernelWidth(N)/2);
            SearchAreaRX=round(validCenterX+ud.SearchWidth(N)+ud.KernelWidth(N)/2);
            SearchAreaTY=round(validCenterY-ud.SearchHeight(N)-ud.KernelHeight(N)/2);
            SearchAreaBY=round(validCenterY+ud.SearchHeight(N)+ud.KernelHeight(N)/2);

           % disp([validCenterX, validCenterY, SearchAreaLX, SearchAreaRX, SearchAreaTY,
SearchAreaBY]);

            %SearchAreaLX=max(round(center(n-1,2*N-1)-ud.SearchWidth(N)-ud.KernelWidth(N)/2),1);
% left boundary
            %SearchAreaRX=min(round(center(n-1,2*N-
1)+ud.SearchWidth(N)+ud.KernelWidth(N)/2),maxX);  % right boundary
            %SearchAreaTY=max(round(center(n-1,2*N)-ud.SearchHeight(N)-ud.KernelHeight(N)/2),1);    %
top boundary
            %SearchAreaBY=min(round(center(n-1,2*N)+ud.SearchHeight(N)+ud.KernelHeight(N)/2),maxY);
% bottom boundary


            SearchArea=double(currentFrame(SearchAreaTY:SearchAreaBY,SearchAreaLX:SearchAreaRX));
% extract search area from frame
            if ud.Field(N)==1      % invert signal in case of brightfield
```

```
            SearchArea=255-SearchArea;
        end
        if ud.Filter(N)==2     % apply threshold filter if selected
            SearchArea(SearchArea<mean2(SearchArea))=0;
        end
        CrossCorrelation=normxcorr2(kernel,SearchArea);
        % find correlation maximum
        [i,j]=find(CrossCorrelation==max(max(CrossCorrelation)));
        if (length(i)==1 & length(j)==1)                % check if there is a unique peak
            %Xleft=max(SearchAreaLX+j-size(kernel,2),1);
            %Ytop=max(SearchAreaTY+i-size(kernel,1),1);
            Xleft=median([(SearchAreaLX+j-size(kernel,2)) ,1, size(currentFrame,2)-ud.KernelWidth(N)]);
            Ytop=median([(SearchAreaTY+i-size(kernel,1)) ,1, size(currentFrame,1)-ud.KernelHeight(N)]);
        else
            disp(char(['Warning! Frame ' num2str(n) ' Could not find unique correlation peak for bead '
num2str(N)]))
            Xleft=max(1,SearchAreaLX);
            Ytop=max(1,SearchAreaTY);
        end
        ud.h.CrosscorrelationResults(n-1,N)=max(max(CrossCorrelation));    %store cross-correlation
result
        % matching kernel from best cross-correlation fit
        matchedKernel=currentFrame(Ytop:Ytop+ud.KernelHeight(N), ...
            Xleft:Xleft+ud.KernelWidth(N));
        offset=[Xleft-1; Ytop-1];
        %compute bead center
        if max(CrossCorrelation(:))>ud.XcorrThr     % if cross-correlation is valid, use centroid of kernel
            newCenter=FindCenter(double(matchedKernel), ...
                ud.Method(N),ud.Field(N),ud.Filter(N))+offset;
        else   % otherwise indicate error by using [-1 -1] as coordinates
            newCenter=[-1; -1];
        end

        if n==NFrames | newCenter(1)==-1   % update kernel only for final frame
            kernels{N} = matchedKernel;
        end

    end
    % store results
    center(n,2*N-1:2*N)=newCenter.';
  end
  waitbar(n/NFrames)        % update progress window
end
close(waitb)       %close wait-window

t=toc;
disp([ud.Filename ' - ' num2str(t) ' sec'])
% display calculation time
% store results
ud.h.lastKernels=kernels;
ud.Results=[center avidata];
set(DemoFig,'UserData',ud);


%%
%% Sub-function DisplayResults
```

```
function DisplayResults
std.Color=[0.8 0.8 0.8];
%std.MenuBar='none';
std.NumberTitle='off';
std.BusyAction='queue';
std.Interruptible='off';
%std.Visible='off';
std.Resize='on';
std.HandleVisibility='on';
std.IntegerHandle='off';
% check if windows already exist
h = findobj(allchild(0), 'tag', 'MPTD Results1');
if ~isempty(h)
   MPTResults1=figure(h(1));
else
   MPTResults1 = figure(std, 'Name','MPTD Results 1: First & Last Frame', ...
      'Tag','MPTD Results1');
end
h = findobj(allchild(0), 'tag', 'MPTD Results2');
if ~isempty(h)
   MPTResults2=figure(h(1));
else
   MPTResults2 = figure(std, 'Name','MPTD Results 2: Kernel Information', ...
      'Tag','MPTD Results2');
end
h = findobj(allchild(0), 'tag', 'MPTD Results3');
if ~isempty(h)
   MPTResults3=figure(h(1));
else
   MPTResults3 = figure(std, 'Name','MPTD Results 3: Cross-Correlation Results', ...
      'Tag','MPTD Results3');
end
h = findobj(allchild(0), 'tag', 'MPTD Results4');
if ~isempty(h)
   MPTResults4=figure(h(1));
else
   MPTResults4 = figure(std, 'Name','MPTD Results 4: Centroid Positions', ...
      'Tag','MPTD Results4');
end
h = findobj(allchild(0), 'tag', 'MPTD Results5');
if ~isempty(h)
   MPTResults5=figure(h(1));
else
   MPTResults5 = figure(std, 'Name','MPTD Results 5: Avi-Data', ...
      'Tag','MPTD Results5');
end

% access results
DemoFig = gcbf;
ud = get(DemoFig, 'UserData');
center=ud.Results(:,1:end-1);
avidata=ud.Results(:,end);
CrossCorrelation=ud.h.CrosscorrelationResults;
Nbeads=ud.Beads;
File=fullfile(ud.Pathname,ud.Filename);
NFrames=size(center,1);    %get number of frames from results data (in case of partially processed files)
```

```
% DISPLAY first and last IMAGE
avi=aviread(File,1);
firstFrame=avi.cdata;
avi=aviread(File,NFrames);
lastFrame=avi.cdata;
clear avi

figure(MPTResults1)
subplot(2,1,1)
imagesc(firstFrame)
colormap(gray)
axis('image')
title('First Frame')
for N=1:Nbeads
   text(center(1,N*2-1),center(1,N*2),num2str(N),'Color','r')
end
subplot(2,1,2)
imagesc(lastFrame)
colormap(gray)
axis('image')
title('Last Frame')
for N=1:Nbeads
   text(center(end,N*2-1),center(end,N*2),num2str(N),'Color','r')
end

% DISPLAY KERNEL
figure(MPTResults2)
clf
for n=1:Nbeads
   firstKernel=ud.h.firstKernels{n};          % retrieve kernel information from kernels
   lastKernel=ud.h.lastKernels{n};
   subplot(2,Nbeads,n)
   imagesc(firstKernel)
   axis('image'); title(char(['Kernel ' num2str(n)]))
   colormap(gray)
   subplot(2,Nbeads,Nbeads+n)
   imagesc(lastKernel)
   axis('image'); title(char(['Kernel ' num2str(n)]))
   colormap(gray)
end
subplot(2,Nbeads,round(Nbeads/2))
xlabel('First Frame')
subplot(2,Nbeads,round(Nbeads*3/2))
xlabel('Last Frame')

% PLOT CROSS-CORRELATION PEAK RESULTS
figure(MPTResults3)
steps=[1:NFrames-1];
for N=1:Nbeads
   subplot(Nbeads,1,N)
   plot(steps,CrossCorrelation(:,N))
   text(1.2,0.1,['Bead ' num2str(N)])
   axis([1 NFrames-1 0 1.1])
   %xlabel('step'); ylabel('peak cross-correlation')
end
```

```
subplot(Nbeads,1,Nbeads); xlabel('step')
subplot(Nbeads,1,round(Nbeads/2)); ylabel('peak cross-correlation')

figure(MPTResults4)
clf
frame=[1:NFrames];
subplot(2,1,1)
hold on
for N=1:Nbeads
   plot(frame,center(:,2*N-1),'bo',frame,center(:,2*N-1),'r:')
   text(frame(end),center(end,2*N-1),['Bead ' num2str(N)]);
end
hold off
title('Bead Center x-position')
xlabel('frame')
ylabel('pixel')

subplot(2,1,2)
hold on
for N=1:Nbeads
   plot(frame,center(:,2*N),'bo',frame,center(:,2*N),'r:')
   text(frame(end),center(end,2*N),['Bead ' num2str(N)]);
end
hold off
title('Bead Center y-position')
xlabel('frame')
ylabel('pixel')

figure(MPTResults5)
clf
frame=[1:NFrames];
% reference for 1 Hz signal
%test=max(avidata)/2+max(avidata)/2*sin((frame-61)/60*2*pi);
%test(1:60)=0;
%test(543:end)=0;
%plot(frame,avidata,'g',frame,test,'r')
plot(frame,avidata)
axis([1 NFrames 0 max(1,1.1*max(avidata))])
title('Magnetic Field')
xlabel('frame')
ylabel('Gauss')
figure(MPTResults1)

%%
%% Sub-function EditParameter
function EditParameter
DemoFig = findobj(allchild(0), 'tag', 'MPTD Program');          % find handle for main menu
ud = get(DemoFig, 'UserData');
% input dialog for program parameters
prompt={'Enter minimum cross-correlation threshold (0-1)'};
def={num2str(ud.XcorrThr)};
dlgTitle='Edit Program Parameter';
lineNo=1;
answer=inputdlg(prompt,dlgTitle,lineNo,def);
if length(answer)
   ud.XcorrThr=median([0 str2num(answer{1}) 1]);    % make sure that value is between 0 and 1
```

```
   set(DemoFig,'UserData',ud);
end
```

Strain analysis program "NSP"

```
% nuclear stretch program
% Jan Lammerding
% 11-10-02
% based on traction force microscopy program
% to do:
% import/export results:
% combine with nuclear strain analysis
% find measure for error on strains

% split into two parts (NSPa and NSPb) on 6/1/03 to compensate for changes
% in MATLAB Release 13

clear

[filename1, pathname]=uigetfile('*.*','Select image 1 (pre-stretch)');
if ~filename1
   return
end

disp('** Register images **')
disp('Select corresponding points in both images, then save points.')
%disp('Hit <return> when done')
disp(['image 1 (unstretched): ' filename1])
file1=fullfile(pathname,filename1);
%img{1}=imread(file1);
im1=imread(file1);
cd(pathname)
figure(1)
imagesc(im1), axis image, colormap gray, title(['base image: ' filename1])

filename='default';
%n=1;
%filenameN{1}=filename1;
stretchN(1,1)=0;
stretch_xyN(1,:)=[0 0];
input_pts_corrN=[];
base_ptsN=[];


%while filename
[filename, pathname]=uigetfile('*.*',['Select stretched image ']);
if filename
   %       n=n+1;
   disp(['image 2 (stretched)  : ' filename])
   file=fullfile(pathname,filename);
   %filenameN{n}=filename;
   im2=imread(file);

   % adjust image intensities
   im1=imadjust(im1,stretchlim(im1,0));
```

```
  im2=imadjust(im2,stretchlim(im2,0));

  % register images:
  %     if n==2        % if this is the first time we use cpselect
  h=cpselect(im2,im1);
  set(h,'WindowClosedCallback','NSPb');
  %     else
  %         h=cpselect(im2,im1, input_pts_corrN{n-1},base_ptsN{n-1});
  %      end
  %      input('Hit <return> to continue ...')
  disp('When done, save points and close window. Results will be computed automatically using
<NSPb>');
  %dispose(h)
end

% nuclear stretch program
% Jan Lammerding
% 11-10-02
% based on traction force microscopy program
% to do:
% import/export results:
% combine with nuclear strain analysis
% find measure for error on strains

% split into two parts (NSPa and NSPb) on 6/1/03 to compensate for changes
% in MATLAB Release 13

% stretchN(1,1)=0;
% stretch_xyN(1,:)=[0 0];
% input_pts_corrN=[];
% base_ptsN=[];
% store points:
input_pts=input_points;
base_pts=base_points;
%clear input_points base_points;

% improve input points using cross-correlation
input_pts_corr=cpcorr(input_pts, base_pts, im2, im1);
% compute transformation based on control point pairs
mytform=cp2tform(input_pts_corr, base_pts, 'linear conformal');
mytform2=cp2tform(input_pts_corr, base_pts, 'affine');
reg_im2=imtransform(im2,mytform,'Xdata',[1 size(im1,2)],'Ydata',[1 size(im1,1)]);
figure(2), imagesc(reg_im2), axis image, colormap gray, title(['input image: ' filename ' - registered'])

% extract the strain-factor from the transformation matrix:
stretch=mytform.tdata.Tinv(1,1);
stretch_xy=[mytform2.tdata.Tinv(1,1) mytform2.tdata.Tinv(2,2)];

% display results:
disp('**************************')
disp('***      Results     ***')
disp('**************************')
disp(sprintf('Image: \t\t%s', filename))
disp(sprintf('Strain: \t%7.3f global', stretch))
disp(sprintf('Strain: \t%7.3f x\t%7.3f y', stretch_xy))
disp(sprintf('Points: \t%3d', size(base_pts,1)))
```

```
disp('Transformation Tensors:')
disp([mytform.tdata.Tinv mytform2.tdata.Tinv])
% store results:
stretchN(1,1)=stretch;
stretch_xyN(1,:)=stretch_xy;
%input_pts=[];
input_pts_corrN=input_pts_corr;
base_ptsN=base_pts;
```

Cytoskeletal displacement tracking program "CSKtracker"

```
% program to  transverse CSK movement in cardiac myocytes
% use n spots centered around bead and cross-correlation algorithm to find
% displacements from frame to frame
%
% cross-correlation algorithm based on MATLAB "cpcorr"-function
% uses cpcorrJan function file
%
% Jan Lammerding
% April 24, 2004
% last edited 4/26/04
% default values:   R=20 pixel, CorrSize=5, Np=6
clear
ud.avipath=cd;
ud.olpath=cd;
ud.savepath=cd;

% start some loop here
filename='default';
while filename
   cd(ud.avipath)    %set path for avi-directory
   [filename, pathname]=uigetfile('*.avi','Select avi-file');
   if filename
      file=fullfile(pathname,filename);
      avi=aviread(file,1);
      infoavi=aviinfo(file);
      % update Userdata
      img=double(avi.cdata);
      ud.avipath = pathname;   % store this path as default for avi-directory
      ud.Pathname=pathname;
      ud.Filename=filename;
      ud.Frames=infoavi.NumFrames;     %number of frames
      figure(1)
      imagesc(img), colormap gray, axis image, title('Click on bead center')
      %display msgbox with avi-file information
      information=char(' FILE INFORMATION','----------------',...
         ['file name:       ' infoavi.Filename], ...
         ['last modified:   ' infoavi.FileModDate],...
         ['file size:       ' num2str(infoavi.FileSize)], ...
         ['number of frames: ' num2str(infoavi.NumFrames)]);
      %msgbox(information)
      %cd(ud.oldpath)  % reset old path

      % select bead center:
      bsize = 20; % size of kernel for bead (optimized for 30x)
```

```
filter = 2; % use threshold filter for bead center determination
csize = 3;  % size of cross to mark bead center
[x, y] = ginput(1);
bx = round(median([1+bsize/2 x size(img,2)-bsize/2]));
by = round(median([1+bsize/2 y size(img,1)-bsize/2]));
% compute bead center
KernelData =img(by-bsize/2:by+bsize/2, bx-bsize/2:bx+bsize/2);
KernelData=max(max(KernelData))-KernelData;          %  invert image for bright field images
if filter==2                                                                    % Threshold filter
   T=mean2(KernelData);                        % use mean as Threshold
   KernelData(KernelData<T)=0;                 % set values below threshold to zero
end
% use estimate for center:
[h w]=size(KernelData);                                          % determine size of kernel

% CENTER OF MASS FIT
[x,y] = meshgrid(1:w,1:h);
bxc = sum(sum(x.*KernelData))/(sum(sum(KernelData)+eps));    % compute center of intensity
byc = sum(sum(y.*KernelData))/(sum(sum(KernelData)+eps));
figure(2)
imagesc(KernelData), axis image, colormap gray, title('Bead center')
bc2=line([bxc-csize bxc+csize bxc bxc bxc], [byc byc byc byc-csize byc+csize]);
set(bc2,'linewidth',2,'color','r')
bxc = bxc + bx - bsize/2;
byc = byc + by - bsize/2;
bc=line([bxc-csize bxc+csize bxc bxc bxc], [byc byc byc byc-csize byc+csize]);
set(bc,'linewidth',2,'color','r')



% compute points on CSK around bead
Np=6;
R=20;
corrsize=5; % half size of correlation window
angInc=2*pi/Np;
ang=([0:Np-1])*angInc;
CSKx = bxc + R*cos(ang);
CSKy = byc + R*sin(ang);
figure(1)
for i=1:Np
   CSKnum(i)=i;
   CSKlabel(i)=text(CSKx(i)+corrsize,CSKy(i),num2str(CSKnum(i)));
   CSKp(i)=line([CSKx(i)-corrsize CSKx(i)+corrsize CSKx(i)+corrsize CSKx(i)-corrsize CSKx(i)-
corrsize], ...
      [CSKy(i)-corrsize CSKy(i)-corrsize CSKy(i)+corrsize CSKy(i)+corrsize CSKy(i)-corrsize]);
   set(CSKp(i),'linewidth',2,'color','b')
end


% now start tracking algorithm
lastframe=uint8(img);
CSKresults=zeros(length(CSKx),2,ud.Frames);
basepoints = [CSKx.' CSKy.'];
base=img;% open next image
CSKresults(:,:,1)=basepoints;
waitb=waitbar(1/ud.Frames,' CSK tracking in progress, please wait...');
```

```
    magField=zeros(ud.Frames,1);
    magField(1)=double(img(1,1));  % read magnetic field values
    for n = 2:ud.Frames
      avi=aviread(file,n);                    % open avi-file at current frame
      currentFrame=avi.cdata;                 % get current frame from avi-file
      magField(n)=double(currentFrame(1,1));     % read magnetic field values from top left pixel
      inputpoints=CSKresults(:,:,n-1);          % use last points as starting points for new search
      newpoints=cpcorr(inputpoints,basepoints,currentFrame,base); % compute new points using cross-
correlation
      %update image and base points;
      %base=currentFrame;
      %basepoints=newpoints;
      CSKresults(:,:,n)=newpoints;
      waitbar(n/ud.Frames)        % update progress window
    end
    close(waitb)       %close wait-window


    % compute displacements:
    % scale=0.3; % microns/pixel, roper 30x
    fps=60; % frame rate
    t=[0:ud.Frames-1]./fps;
    CSKxDisp = zeros(ud.Frames,length(CSKnum));
    CSKyDisp = zeros(ud.Frames,length(CSKnum));
    lgd=['magnetic field'];
    figure(3)
    for i=1:length(CSKnum)
      xdisp=squeeze(CSKresults(i,1,:));
      ydisp=squeeze(CSKresults(i,2,:));
      CSKxDisp(:,i)=xdisp-mean(xdisp(1:fps));
      CSKyDisp(:,i)=ydisp-mean(ydisp(1:fps));
      lgd=char(lgd,['region ' num2str(i)]);
      subplot(Np,1,i)
      plot(t,CSKxDisp(:,i)), ylabel(['Region ' num2str(i)])
    end
    meanCSKxDisp=mean(CSKxDisp,2);
    meanCSKyDisp=mean(CSKyDisp,2);
    disp('*****************************')
    disp(['File: ' filename])
    figure(4)
    plot(t,magField/max(magField),t,CSKxDisp)
    legend(lgd), xlabel('time'), ylabel('pixel'), title('CSK-Displacement in x-direction')
    figure(5)
    plot(t,magField/max(magField),t,CSKyDisp)
    legend(lgd), xlabel('time'), ylabel('pixel'), title('CSK-Displacement in y-direction')
    figure(6)
    lgd2=char('magnetic field','mean CSK x-displacement','mean CSK y-displacement');
    plot(t,magField/max(magField),t,meanCSKxDisp,t,meanCSKyDisp)
    legend(lgd2), xlabel('time'), ylabel('pixel')

    disp('press key to continue')
    pause
    % compute corrected mean, i.e. ignoring invalid results
    outlier=inputdlg('Enter outlier you wish to ignore','CSK tracking program',1,{'0'});
    outlier=char(outlier);%outlier=str2num(outlier{1});
    validmeanX = [];
```

```
    validmeanY = [];
    for i = 1:size(CSKxDisp,2)
      if max(CSKxDisp(:,i))==0 | findstr(num2str(i), outlier)   % if the region has a valid displacement
and was not selected as an outlier
        disp(['Region ' num2str(i) ': excluded'])
      else
        validmeanX = [validmeanX CSKxDisp(:,i)];
        validmeanY = [validmeanY CSKyDisp(:,i)];
      end
    end
    validmeanX = mean(validmeanX,2);
    validmeanY = mean(validmeanY,2);
    figure(6)
    lgd2=char('magnetic field','mean CSK x-displacement','mean CSK y-displacement');
    plot(t,magField/max(magField), t, validmeanX, t, validmeanY)
    legend(lgd2), xlabel('time'), ylabel('pixel'), title('valid means - ignoring zero-displacement regions')

    % save results as csv-file
    cd(ud.savepath);
    [path, name, ext, ver]=fileparts(filename);
    initfile=[name '_CSKresults-R' num2str(R) '.csv'];
    [sfilename, pathname]=uiputfile(initfile,'Save Results to');
    if sfilename                          % save results if user selected filename
      ud.savepath=pathname;
      [pathstr, fname, ext, vers]=fileparts(sfilename);
      if ~length(ext)
        filename=[filename '.csv'];
      end
      % create results in correct format
      ud.Results=[];
      ud.Results=[validmeanX validmeanY magField];
      file=fullfile(pathname,sfilename);
      csvwrite(file,ud.Results);          % write results to csv-file
    end
  end
end
```

Strain mapping program "SMPavi"

```
function ud = SMP(action, varargin)
% Strain Mapping Program with GUI to analyze AVI-movies
% and store results as avi-files
% Jan Lammerding
% 11/28/2001
% updated save movie function, start it automatically after avi-generation
% changed display for displacement vectors
% fixed load result function
% 11/29/2001:
% only display valid vectors
% 02/06/2002:
% include intensity threshold filter
% 02/08/2002:
% use intensity, stdv and max-corr filter for postprocessing
% add option to display stdv results
% allow user to skip frames
```

```
% removed option for sub-pixel resolution
% 02/08/2002
% allow polygonal region of interest
% express displacements in micron (allow scale input)
% 02/15/2002
% update help text
% fix display of ROI after loading Results
% add (press <h> for help) - text
% update saveResult function: don't save movie
% correct display of frame number
% 02/17/2002
% fixed incorrect result display when window was closed
% fixed ROI problem by reverting back to old loadResult structure instead of calling updateROI
% fixed problem for zero displacements by adding eps to [minD maxD]
% replaced ud.h.rect with ud.h.ROIhandle
% 10/01/2002
% added option to measure relative or absolute displacements
% 10/04/2002
% add Erase movie function to preserve memory
% add option to manualy set color bar limits
% allow user to abort display
% added rel. or abs difference for image difference
% fixed save as CSV function
% fixed image difference
% fixed ROI error when loading results
% 10/05/2002
% removed automatic save movie call after result display
% left to do:
% add display of single result frames
% add option for FFT cross-correlation (especially phase)

if nargin<1,
   action='InitializeSMP';
   if nargout>0  % If user wants the UserData, give it to them.
      ud = feval(action,varargin{:});
      return
   end
end

feval(action,varargin{:})
return


%%
%% subfunction: initialize Strain Mapping Program (SMPavi)
function udout = InitializeSMP()
% If SMPavi is already running, bring it to the foreground
h = findobj(allchild(0), 'tag', 'SMPavi Program');
if ~isempty(h)
   figure(h(1))
   if nargout>0
      udout = get(h(1), 'UserData');
   end
   return
end
```

```
SMPFig = figure('Color',[0.8 0.8 0.8], ...
   'Colormap', gray(256), ...
   'MenuBar','figure', ...
   'BusyAction','Queue','Interruptible','off',...
   'Name','Strain Mapping Program', ...
   'Tag','SMPavi Program',...
   'Visible','off','Resize','off','HandleVisibility', 'on',...
   'NumberTitle','off', 'IntegerHandle', 'off', ...
   'Position',[100 180 840 425], ...
   'KeyPressFcn','SMPavi("KeyPress");',...
   'CloseRequestFcn','SMPavi("CloseSMP");');
% initiliaze user data
ud.FigureHandle = SMPFig;
ud.ImageFile='';
ud.infoavi=[];
ud.ImageData{1}=[]; ud.ImageData{2}=[];
ud.Pathname=pwd;
ud.MinMax=[];
ud.SearchWidth=10;
ud.SearchHeight=10;
ud.KernelWidth=20;
ud.KernelHeight=20;
ud.IncrementHor=10;
ud.IncrementVer=10;
ud.XcorrMaxThreshold=0.5;
ud.StdMin=5;
ud.skipFrame=1;
ud.MeanIntThreshold=1;
ud.ROI=[];
ud.ROIpoly=[];
ud.ROImask=[];
%ud.Scale=0.1611; %for coolsnap 40x
ud.Scale=0.3; % for Roper ES310T 30x
ud.Pause='no';
ud.RelDisp=1;
ud.Results=[];
ud.Movie=[];
ud.MovieTitle=[];
ud.CbarMin=NaN;
ud.CbarMax=NaN;
% handles ROI marker and current image indicator
ud.h.ROIhandle=[];
ud.h.CurrentImage=1;
ud.t=[];
ud.t.skipFrame=[];
% define standard settings
Std.Interruptible = 'off';
Std.BusyAction = 'queue';
% Defaults for image axes
Ax = Std; Ax.Units = 'Pixels';
Ax.Parent = SMPFig; Ax.XTick = [];
Ax.YTick = []; Ax.Box='on';
Img = Std; Img.CData = [];
%Img.CDataMapping = 'Scaled';
Ctl = Std; Ctl.Units = 'Pixels';
Ctl.Parent = SMPFig;
```

```
Btn=Ctl; Btn.Style = 'pushbutton';
Frame = Ctl; Frame.Style = 'Frame';
Edit = Ctl; Edit.Style = 'edit';
Edit.HorizontalAlignment = 'right';
Edit.BackgroundColor = 'white';
Edit.ForegroundColor = 'black';
Menu = Ctl; Menu.Style = 'Popupmenu';
Text = Ctl; Text.Style = 'text'; Text.BackgroundColor=[0.7 0.7 0.7];
Text.HorizontalAlignment = 'left';
bgd=[0.7 0.7 0.7];              % background
%Image Menu
Frame1 = uicontrol(Frame, ...
   'BackgroundColor',bgd,...
   'Position',[575 285 255 120]);
ud.h.DisplayImage1 = uicontrol(Btn, ...
   'Position',[[590 365 150 30]], ...
   'String','Frame -',...
   'Callback','SMPavi("DisplayImage")');
ud.h.DisplayImage2 = uicontrol(Btn, ...
   'Position',[590 325 150 30], ...
   'String','Frame +',...
   'Callback','SMPavi("DisplayImage")');
ud.h.BrowseFigure1 = uicontrol(Btn, ...
   'Position',[765 335 50 50], ...
   'String','Open avi', ...
   'Callback','SMPavi("BrowseFigure")');
h1 = uicontrol(Text, ...
   'FontWeight','bold', ...
   'Position',[590 293 30 20], ...
   'String','min');
ud.h.SetMin = uicontrol(Edit, ...
   'Position',[615 295 30 20], ...
   'String','0', ...
   'Callback','SMPavi("SetMinMax")');
h1 = uicontrol(Text, ...
   'FontWeight','bold', ...
   'Position',[665 293 30 20], ...
   'String','max');
ud.h.SetMax = uicontrol(Edit, ...
   'Position',[690 295 30 20], ...
   'String','255', ...
   'Callback','SMPavi("SetMinMax")');
ud.h.Histogram = uicontrol(Btn, ...
   'Position',[740 295 75 20], ...
   'String','Histogram', ...
   'Callback','SMPavi("Histogram")');
%Perform CrossCorrelation
ud.h.CrossCorrelation = uicontrol(Btn, ...
   'Position',[600 245 205 25], ...
   'String','Cross-Correlation', ...
   'Callback','SMPavi("CrossCorrelationMenu")');
% Result Menu
Frame1 = uicontrol(Frame, ...
   'BackgroundColor',bgd,...
   'Position',[575 105 255 125]);
ud.h.LoadResults = uicontrol(Btn, ...
```

```
   'Position',[590 188 100 30], ...
   'String','Load Results', ...
   'Callback','SMPavi("LoadResults")');
ud.h.SaveResults = uicontrol(Btn, ...
   'Position',[715 188 100 30], ...
   'String','Save Results', ...
   'Callback','SMPavi("SaveResults")');
ud.h.ResultType = uicontrol(Menu, ...
   'Position',[590 150 100 25], ...
   'String',{'displ. vectors','x-displacement','y-displacement','abs-displ.',...
      'x-corr max.','x-corr. error','standard dev.','image difference'}, ...
   'Value',4,...
   'Callback','SMPavi("UpdateInterpol");');
ud.h.Interpolation = uicontrol(Menu, ...
   'Position',[715 150 100 25], ...
   'String',{'no interpolation','linear interp.','nearest interp.','cubic interp.'}, ...
   'Value',2);
ud.h.DisplayResults = uicontrol(Btn, ...
   'Position',[615 113 175 30], ...
   'String','Display Results',...
   'Callback','SMPavi("DisplayResults")');
% ResultsTextWindow
Frame1 = uicontrol(Frame, ...
   'BackgroundColor',bgd,...
   'Position',[[575 25 255 75]]);
ud.h.FigureString = uicontrol(Text, ...
   'FontWeight','bold', ...
   'BackgroundColor',bgd,...
   'Position',[590 80 225 15], ...
   'String','AVI-file: ');
ud.h.MovieString = uicontrol(Text, ...
   'FontWeight','bold', ...
   'BackgroundColor',bgd,...
   'Position',[590 60 225 15], ...
   'String','');
HelpText = uicontrol(Text, ...
   'FontWeight','normal', ...
   'BackgroundColor',bgd,...
   'Position',[650 30 120 15], ...
   'String','Press <h> for help');
% ImageDisplay
StartImg=200*ones(480,640);   %StartImg(220:260,300:340)=250*rand(41,41);
ud.h.ImageAxes = axes(Ax, ...
   'Position', [37 24 500 375]);
ud.h.ImageImage = image(StartImg, Img,...
   'Parent',ud.h.ImageAxes, ...
   'XData',[1 size(StartImg,2)],'YData',[1 size(StartImg,1)]);
set(ud.h.ImageImage,'XData',[1 640],'YData',[1 480]);
set(ud.h.ImageAxes,'DataAspectRatio',[1 1 1],'TickDir','in');
%update and display SMPavi figure
set([ud.h.CrossCorrelation ud.h.SaveResults ud.h.DisplayResults],'enable','off')
set(SMPFig, 'Userdata', ud);
set(SMPFig, 'visible','on');

if nargout>0
   udout = ud;
```

```
end
return


%%
%% Sub-function KeyPress
function KeyPress
DemoFig = gcbf;
ud = get(DemoFig, 'UserData');
key = real(get(DemoFig, 'CurrentCharacter'));
if isempty(key), % switch statement was triggering error in Unix, the
   return;      % "Shift" key was causing the variable key to be []
end
switch key
case {8, 127}  % delete and backspace keys
   %PreviousBead                    % same as prev-bead
case {13, 3}   % enter and return keys
   %NextBead
case {102, 70}  % 'f' or 'F'
   EditFilter
case {104, 72}  % 'h' or 'H'
   HelpWindow  %
case {115, 83} % 's' or 'S'
   SaveMovie
case {112, 80} % 'p' or 'P'
   EditParameter
case {101, 69}  %'e' or 'E'
   EraseMovie
case 27                         % escape key
   CloseSMP
end

%%
%% Sub-Function - CloseSMP
function CloseSMP
% Close request function for the SMPavi program
% User-defined close request function
% to display a question dialog box
selection = questdlg('Do you really want to quit ?',...
   'Quit SMPavi Request',...
   'Yes','No','Yes');
switch selection,
case 'Yes',
   delete(gcf)
   for i=12:18
      if ishandle(i)
         close(i)
      end
   end
case 'No'
   return
end

%%
%% Sub-Function BrowseFigure
function BrowseFigure
```

```
DemoFig = gcbf;
ud = get(DemoFig, 'UserData');
cd(ud.Pathname)  % added this 9/22/01
[filename, pathname]=uigetfile('*.avi','Load avi-file ');
if filename
   set(DemoFig,'Pointer','watch')
   file=fullfile(pathname,filename);
   avi=aviread(file,1);
   ud.infoavi=aviinfo(file);
   % update Userdata
   img=avi.cdata;
   %clear avi
   avi=aviread(file,2);
   img2=avi.cdata;
   clear avi
   minimum=double(min(img(:)));
   maximum=double(max(img(:)));
   % update Userdata
   ud.ImageFile=filename;
   ud.ImageData{1}=img;
   ud.ImageData{2}=img2;
   ud.Pathname=pathname;
   ud.MinMax=[minimum maximum];
   set(ud.h.SetMin,'String',num2str(minimum));
   set(ud.h.SetMax,'String',num2str(maximum));
   ud.ROI=[1 1 size(img')];          % set region of interest to entire image
   if (size(ud.ImageData{1}) & size(ud.ImageData{2}))                % enable cross-correlation if both
images have been loaded.
      set(ud.h.CrossCorrelation,'enable','on')
   end
   ud.h.CurrentImage=1;
   set(ud.h.ImageImage, 'Cdata', img,...
      'XData',[1 size(img,2)],'YData',[1 size(img,1)]);
   set(ud.h.ImageAxes,'XLim',[1 size(img,2)],'YLim',[1 size(img,1)]);
   set(ud.h.FigureString,'String',['AVI-file: ' ud.ImageFile ' - frame 1']);
   set(DemoFig,'UserData',ud);
   set(DemoFig,'Pointer','arrow')
end

%%
%% Sub-function Histogram
function Histogram
DemoFig = gcbf;
ud = get(DemoFig, 'UserData');
%fig=ud.h.CurrentImage;
if size(ud.ImageData{1})
   figure(10)
   set(10,'Name','Histogram','NumberTitle','off')
   imhist(ud.ImageData{1})
   title(['Histogram for ' ud.ImageFile])
end

%%
%% Sub-function DisplayImage
function DisplayImage
DemoFig = gcbf;
```

```
ud = get(DemoFig, 'UserData');
switch gco;                    % find out which image button was pressed
case ud.h.DisplayImage1
   fig=max(ud.h.CurrentImage-1,1);   % one frame backwards
case ud.h.DisplayImage2
   fig=min(ud.h.CurrentImage+1,ud.infoavi.NumFrames);
otherwise
   fig=ud.h.CurrentImage;
   disp('something is wrong here in Histogram')
   %return;
end
file=fullfile(ud.Pathname,ud.ImageFile);
avi=aviread(file,fig);
img=avi.cdata;
clear avi
img(img<ud.MinMax(1))=0;
img(img>ud.MinMax(2))=255;
set(ud.h.ImageImage, 'Cdata', img);
set(ud.h.FigureString,'String',['AVI-file: ' ud.ImageFile ' - frame ' num2str(fig)]);
ud.h.CurrentImage=fig;
set(DemoFig,'UserData',ud);

%%
%% Sub-function SetMinMax
function SetMinMax
DemoFig = gcbf;
ud = get(DemoFig, 'UserData');
fig=ud.h.CurrentImage;
newMin=max(str2num(get(ud.h.SetMin,'String')),0);
newMax=min(str2num(get(ud.h.SetMax,'String')),255);
set(ud.h.SetMin,'String',num2str(newMin));
set(ud.h.SetMax,'String',num2str(newMax));
ud.MinMax=[newMin newMax];
if size(ud.ImageData{fig})
   img=ud.ImageData{fig};
   img(img<ud.MinMax(1))=0;
   img(img>ud.MinMax(2))=255;
   set(ud.h.ImageImage, 'Cdata', img);
   set(DemoFig,'UserData',ud);
end

%%
%% sub-function cross-correlation menu
function CrossCorrelationMenu
DemoFig = gcbf;
ud = get(DemoFig, 'UserData');
%% create crosscorrelation menu
XcorrelationMenu = figure('Color',[0.7 0.7 0.7], ...
   'Colormap', gray(256), ...
   'MenuBar','none', ...
   'BusyAction','Queue','Interruptible','off',...
   'Name','Cross-Correlation Menu', ...
   'Tag','CrossCorrelationMenu',...
   'Visible','off','Resize','off','HandleVisibility', 'on',...
   'NumberTitle','off', 'IntegerHandle', 'off', ...
   'Position',[585 348 251 334], ...
```

```
   'CloseRequestFcn','SMPavi("CancelXcorrelation");');
Std.Interruptible = 'off';
Std.BusyAction = 'queue';
% Defaults for image axes
Ctl = Std; Ctl.Units = 'Pixels';
Ctl.Parent = XcorrelationMenu;
Btn=Ctl; Btn.Style = 'pushbutton';
Edit = Ctl; Edit.Style = 'edit';
Edit.HorizontalAlignment = 'right';
Edit.BackgroundColor = 'white';
Edit.ForegroundColor = 'black';
Text = Ctl; Text.Style = 'text'; Text.BackgroundColor=[0.7 0.7 0.7];
TextBold=Text; TextBold.HorizontalAlignment = 'left'; TextBold.FontWeight = 'bold';
Checkbox=Ctl; Checkbox.Style='Checkbox'; Checkbox.BackgroundColor= [0.7 0.7 0.7];
h1 = uicontrol(Text, ...
   'Position',[101 306 50 20], ...
   'String','horizontal');
h1 = uicontrol(Text, ...
   'Position',[176 306 50 20], ...
   'String','vertical');
h1 = uicontrol(Text, ...
   'Position',[21 276 75 30], ...
   'String','maximal displacement');
ud.t.maxDisplHor = uicontrol(Edit, ...
   'Position',[101 281 50 20], ...
   'String',num2str(ud.SearchWidth),...
   'Callback','SMPavi("UpdateParameters")');
ud.t.maxDisplVer = uicontrol(Edit, ...
   'Position',[176 281 50 20], ...
   'String',num2str(ud.SearchHeight),...
   'Callback','SMPavi("UpdateParameters")');
h1 = uicontrol(Text, ...
   'Position',[21 236 75 30], ...
   'String','x-correlation window');
ud.t.XcorrHor = uicontrol(Edit, ...
   'Position',[101 241 50 20], ...
   'String',num2str(ud.KernelWidth),...
   'Callback','SMPavi("UpdateParameters")');
ud.t.XcorrVer = uicontrol(Edit, ...
   'Position',[176 241 50 20], ...
   'String',num2str(ud.KernelHeight),...
   'Callback','SMPavi("UpdateParameters")');
h1 = uicontrol(Text, ...
   'Position',[21 196 75 30], ...
   'String','window increments');
ud.t.incrementHor = uicontrol(Edit, ...
   'Position',[101 201 50 20], ...
   'String',num2str(ud.IncrementHor),...
   'Callback','SMPavi("UpdateParameters")');
ud.t.incrementVer = uicontrol(Edit, ...
   'Position',[176 201 50 20], ...
   'String',num2str(ud.IncrementVer),...
   'Callback','SMPavi("UpdateParameters")');
ud.t.skipFrame = uicontrol(TextBold, ...
   'Position',[21 138 175 30], ...
   'String',['process every ' num2str(ud.skipFrame) ' frame(s)']);
```

```
ud.t.EditFilter= uicontrol(Btn, ...
   'Position',[191 151 40 20], ...
   'String','Edit', ...
   'Callback','SMPavi("EditFilter")');
ud.t.PickROI= uicontrol(Btn, ...
   'Position',[51 56 150 25], ...
   'String','Pick Region of Interest (ROI)', ...
   'Callback','SMPavi("PickROI")');
ud.t.StartXcorrelation = uicontrol(Btn, ...
   'Position',[26 16 125 25], ...
   'String','Start Cross-Correlation', ...
   'Callback','SMPavi("StartXcorrelation")');
ud.t.CancelXcorrelation = uicontrol(Btn, ...
   'Position',[176 16 50 25], ...
   'String','Cancel', ...
   'Callback','SMPavi("CancelXcorrelation")');
set(DemoFig, 'Userdata', ud);
set(XcorrelationMenu, 'visible','on');
UpdateParameters

%%
%% sub-function CancelXcorrelation
function CancelXcorrelation
delete(gcbf)

%%
%% sub-function PickROI
function PickROI
DemoFig = findobj(allchild(0), 'tag', 'SMPavi Program');        % find handle for main menu
ud = get(DemoFig, 'Userdata');
figure(11)
set(11,'Name','Region of Interest','NumberTitle','off')
imagesc(ud.ImageData{1})
colormap gray; axis image
title('Select Region of Interest (left - add point ; right - final point)')
%[BW, xi ,yi]=roipoly(ud.ImageData{1});
[BW, xi ,yi]=roipoly;
extraWidth=ud.KernelWidth+ud.SearchWidth;      % make sure that entire ROI is inside field of view
extraHeight=ud.KernelHeight+ud.SearchHeight;
rect=[min(xi)-extraWidth min(yi)-extraHeight (max(xi)-min(xi)+1)+2*extraWidth (max(yi)-
min(yi)+1)+2*extraHeight];
bound=[1 1 size(ud.ImageData{1}')];
rect=[max(rect(1:2),bound(1:2)) min(rect(3:4),bound(3:4))]; % make sure that it is within image
close(11)
ud.ROI=round(rect);
ud.ROIpoly=[xi yi];
ud.ROImask=BW;
set(DemoFig,'UserData',ud);
UpdateROI

%%
%% Sub-function UpdateROI
function UpdateROI
DemoFig = findobj(allchild(0), 'tag', 'SMPavi Program');        % find handle for main menu
ud = get(DemoFig, 'UserData');
% rectangle to indicate ROI
```

```
delete(ud.h.ROIhandle)     % delete old rectangle
%RectX=[ud.ROI(1) ud.ROI(1)+ud.ROI(3) ud.ROI(1)+ud.ROI(3) ud.ROI(1) ud.ROI(1)];
%RectY=[ud.ROI(2) ud.ROI(2) ud.ROI(2)+ud.ROI(4) ud.ROI(2)+ud.ROI(4) ud.ROI(2)];
RectX=ud.ROIpoly(:,1);
RectY=ud.ROIpoly(:,2);
ud.h.ROIhandle = line('XData', RectX, ...
   'YData', RectY, ...
   'Parent', ud.h.ImageAxes, ...
   'Clipping', 'on', ...
   'Color', 'r', ...
   'LineStyle', '-', ...
   'EraseMode', 'normal');
set(DemoFig,'UserData',ud);

%%
%% sub-function UpdateParameters
function UpdateParameters
DemoFig = findobj(allchild(0), 'tag', 'SMPavi Program');        % find handle for main menu
ud = get(DemoFig, 'UserData');
ud.SearchWidth=max(str2num(get(ud.t.maxDisplHor,'String')),0);                % mininum: 0
ud.SearchHeight=max(str2num(get(ud.t.maxDisplVer,'String')),0);       % minimum: 0
ud.KernelWidth=max(str2num(get(ud.t.XcorrHor,'String')),2);                % minimum: 2
ud.KernelHeight=max(str2num(get(ud.t.XcorrVer,'String')),2);           % minimum: 2
ud.IncrementHor=max(str2num(get(ud.t.incrementHor,'String')),1);           % mininum: 1
ud.IncrementVer=max(str2num(get(ud.t.incrementVer,'String')),1);           % mininum: 1
%ud.SubPixelResolution=get(ud.t.SubPixelResolution,'Value');
set(DemoFig,'UserData',ud);

%%
%% Sub-function StartXcorrelation
function StartXcorrelation
DemoFig = findobj(allchild(0), 'tag', 'SMPavi Program');        % find handle for main menu
ud = get(DemoFig, 'UserData');
KernelSize=[ud.KernelWidth ud.KernelHeight];
SearchSize=[ud.SearchWidth ud.SearchHeight];
Increments=[ud.IncrementHor ud.IncrementVer];
%subPixel=ud.SubPixelResolution;
rect=ud.ROI;
if ~size(ud.ROIpoly)
   ud.ROImask=uint8(ones(ud.infoavi.Height,ud.infoavi.Width));
end
set(ud.t.StartXcorrelation,'enable','off')
h=waitbar(0,'Cross-correlation in progress ...');
% clear old results
ud.Results=[];
ProcessNframes=floor((ud.infoavi.NumFrames-1)/ud.skipFrame);
%for n=1:(ud.infoavi.NumFrames-1)
for n=1:ProcessNframes
   file=fullfile(ud.Pathname,ud.ImageFile);
   %    avi=aviread(file,n);
   if ud.RelDisp;  % in case of relative displacement
      avi=aviread(file,(n-1)*ud.skipFrame+1);
      im1=avi.cdata;
      im1=immultiply(im1,ud.ROImask);
   else    % absolute displacements
      avi=aviread(file,1);
```

```
    im1=avi.cdata;
    im1=immultiply(im1,ud.ROImask);
  end
  %avi=aviread(file,n+1);
  avi=aviread(file,n*ud.skipFrame+1);
  im2=avi.cdata;
  im2=immultiply(im2,ud.ROImask);
  clear avi
  % apply min/max settings
  im1(im1<ud.MinMax(1))=0;
  im1(im1>ud.MinMax(2))=255;
  im2(im2<ud.MinMax(1))=0;
  im2(im2>ud.MinMax(2))=255;
  % get x-correlation results from StrainmappingFunc-function
  ud.Results{n}=StrainMappingFunc(im1,im2,KernelSize,SearchSize,Increments, rect);
  %waitbar(n/(ud.infoavi.NumFrames-1),h)
  waitbar(n/ProcessNframes,h)
end
close(h)
delete(gcbf)
set(DemoFig,'UserData',ud);
set([ud.h.DisplayResults ud.h.SaveResults],'enable','on');

%%
%% Sub-function StrainMappingFunc
function result=StrainMappingFunc(im1, im2, KernelSize, SearchSize, Increments, rect)
% functions for strain mapping program
% Jan Lammerding
% 8-16-01
% syntax: result=strainmapping(image1, image2, Kernelsize, Searchsize, Increments, region of interest)
% where:
% result 3D-matrix  layer 1 & 2: x- and y- coordinates of result vectors
%             layer 3 & 4: x- and y-displacement of result vectors
%                                          layer 5: cross-correlation value (max)
%             layer 6: error indices for result vectors: 0: no error, 1: comp. displacement too large (only for
subpixel)
%                                      2: x-correlation peak at matrix edge (only for subpixel)
%                                      3: no unique x-correlation peak  4: kernel stdev<StdMin 5: x-
correl peak too small
%             layer 7: mean intensity of kernel
%             layer 8: standard deviation of kernel
KernelWidth=KernelSize(1);  KernelHeight=KernelSize(2);
SearchWidth=SearchSize(1);  SearchHeight=SearchSize(2);
Xincrement=Increments(1); Yincrement=Increments(2);
[maxY,maxX]=size(im1);
minX=max(rect(1)-SearchWidth,1);   % left boundary of ROI
maxX=min(maxX,rect(1)+rect(3)+SearchWidth+KernelWidth); %right boundary of ROI
minY=max(rect(2)-SearchHeight,1);   % top boundary of ROI
maxY=min(maxY,rect(2)+rect(4)+SearchHeight+KernelHeight);
% initialize variables & prepare result output
X=round([minX+SearchWidth:Xincrement:maxX-SearchWidth-KernelWidth]);
Y=round([minY+SearchHeight:Yincrement:maxY-SearchHeight-KernelHeight]);
[x,y]=meshgrid(X,Y);
result=zeros(length(Y),length(X),8);          %result format: 6-stacked matrices
result(:,:,1)=x+KernelWidth/2;
result(:,:,2)=y+KernelHeight/2;
```

```
for i=1:length(X)
  for j=1:length(Y)
    LeftX=x(j,i);
    TopY=y(j,i);
    kernel=im1(TopY:(TopY+KernelHeight),LeftX:(LeftX+KernelWidth));
    %disp(['kernel ' num2str(j) ' / ' num2str(i) ': ' num2str(mean2(kernel))])
    [xs,ys,XcorrMax,ErrorIndex]=Xcorrelation(kernel,im2,LeftX,TopY,SearchWidth,SearchHeight);
    result(j,i,3)=xs;
    result(j,i,4)=ys;
    result(j,i,5)=XcorrMax;
    result(j,i,6)=ErrorIndex;
    % added 2/6/02:
    result(j,i,7)=mean2(kernel);
    result(j,i,8)=std2(kernel);
  end
end

%%
%% Sub-function for cross-correlation
function [Xshift, Yshift, XcorrMax,
ErrorIndex]=Xcorrelation(kernel,im2,LeftX,TopY,SearchWidth,SearchHeight)
[KernelHeight,KernelWidth]=size(kernel);
StartX=LeftX-SearchWidth;                    % starting point x
StartY=TopY-SearchHeight;                    % starting point y
% compute cross correlation c for every point in search area
SearchArea=im2(TopY-SearchHeight:TopY+SearchHeight+KernelHeight-1,LeftX-
SearchWidth:LeftX+SearchWidth+KernelWidth-1);
if std2(kernel)>1  %requires that kernel is non uniform , changed this back to 1 as minimum on 2-7-02
  CrossCorrelation=normxcorr2(kernel,SearchArea);
  %PLOT KERNEL AND SEARCH AREA
  XcorrMax=max(CrossCorrelation(:));
  [i,j]=find(CrossCorrelation==XcorrMax);
  if (length(i)==1 & length(j)==1)          % check if there is a unique peak
    % calculate only integer displacements
    Xshift=j-SearchWidth-KernelWidth;
    Yshift=i-SearchHeight-KernelHeight;
    ErrorIndex=0;
  else
    Xshift=0;
    Yshift=0;
    ErrorIndex=3;
  end
else
  Xshift=0;
  Yshift=0;
  XcorrMax=0;
  ErrorIndex=4;
end

%%
%% Sub-function UpdateInterpol
function UpdateInterpol
% disable interpolation option for certain result displays
DemoFig = gcbf;
ud = get(DemoFig, 'UserData');
DisplayChoice=get(ud.h.ResultType,'Value');          % check which results are requested
```

```
switch DisplayChoice
case 1     % for disp-vectors
   set(ud.h.Interpolation,'String',{'Scale: 1','Scale: 2','Scale: 5','Scale: 10','Scale: 100','Scale: 1/2'})
   set(ud.h.Interpolation,'Value',1)
   set(ud.h.Interpolation,'Enable','on')
case {5,6,7,8}     % for x-corr max, x-corr err., image difference
   set(ud.h.Interpolation,'Enable','off')
otherwise
   set(ud.h.Interpolation,'String',{'no interpolation','linear interp.','nearest interp.','cubic interp.'})
   set(ud.h.Interpolation,'Enable','on')
end

%%
%% Sub-function DisplayResults
function DisplayResults
DemoFig = gcbf;
ud = get(DemoFig, 'UserData');
Nframes=size(ud.Results,2);
% display results
% extract x- and y-shift from results, set vectors with error to 0
[maxY maxX]=size(ud.ImageData{1});
[xi,yi]=meshgrid([ud.Results{1}(1,1,1):ud.Results{1}(1,end,1)],[ud.Results{1}(1,1,2):ud.Results{1}(end,1
,2)]);     %interpolated coordinates
DisplayChoice=get(ud.h.ResultType,'Value');          % check which results are requested
Interpolation=get(ud.h.Interpolation,'Value');  % check if interpolation is requested
switch DisplayChoice
case 1               % Quiver Plot
   scale=[1 2 5 10 100 0.5];
   figure(12)
   set(12,'Name','Displacement
Vectors','NumberTitle','off','CloseRequestFcn','SMPavi("DisplayAbort");','Userdata',0)
   ud.Movie=[];
   for n=1:Nframes
      stop=get(12,'userdata');
      if stop
         break
      end
      if gcf~=12
         figure(12)
         % set(12,'Name','Displacement Vectors','NumberTitle','off')
      end
      x=ud.Results{n}(:,:,1);                              % Result vector x-positon
      y=ud.Results{n}(:,:,2);                              % result vector y-position
      dx=ud.Results{n}(:,:,3);                             % x-displacement
      dy=ud.Results{n}(:,:,4);                             % y-displacement
      XcorrMax=ud.Results{n}(:,:,5);          % cross-correlation maxima
      ErrorIndex=ud.Results{n}(:,:,6);% error index
      % 2-6-02: use filters for result display
      meanInt=ud.Results{n}(:,:,7);  % mean intensity of each kernel
      KernelStd=ud.Results{n}(:,:,8); % kernel standard deviation
      avgMean=mean2(meanInt);
      ErrorIndex(meanInt<avgMean*ud.MeanIntThreshold)=6;                % apply intensity filter
      ErrorIndex(XcorrMax<ud.XcorrMaxThreshold)=5;       % apply x-corr-maxima filter
      ErrorIndex(KernelStd<ud.StdMin)=4;                % apply std-filter

      dx(ErrorIndex>0)=0; % neglect results with error
```

```
      dy(ErrorIndex>0)=0; % neglect results with error

      file=fullfile(ud.Pathname,ud.ImageFile);
      avi=aviread(file,(n-1)*ud.skipFrame+1);
      img=avi.cdata; clear avi;

      clf
      hold on
      imagesc(img), colormap gray
      %       quiver(x(:),y(:),dx(:)*scale(Interpolation),dy(:)*scale(Interpolation),0,'r')
      %quiver(x(:),y(:),dx(:)*scale(Interpolation),dy(:)*scale(Interpolation),0,'r') % plot all vectors
      % only plot valid vectors

quiver(x(~ErrorIndex),y(~ErrorIndex),dx(~ErrorIndex)*scale(Interpolation),dy(~ErrorIndex)*scale(Interpo
lation),0,'r')
      % plot error vectors blue

%quiver(x(ErrorIndex>0),y(ErrorIndex>0),dx(ErrorIndex>0)*scale(Interpolation),dy(ErrorIndex>0)*scale(
Interpolation),0,'b')
      %       axis ij,axis image, axis([1 maxX 1 maxY]);
      axis ij,axis image, axis([ud.Results{1}(1,1,1) ud.Results{1}(1,end,1) ud.Results{1}(1,1,2)
ud.Results{1}(end,1,2)]);
      hold off
      title(['Displacement Vectors (Scale ' num2str(scale(Interpolation)) ')'])
      xlabel(['frame: ' num2str(n) ' / ' num2str(Nframes)])
      M(n)=getframe(12);
      if strcmp(lower(ud.Pause),'yes') & n<Nframes       % pause if ud.Pause is activated and we are not at
last frame
         pause
      end
   end
   if n==Nframes
      set(12,'CloseRequestFcn','delete(gcbf)')
      ud.Movie=M;
      ud.MovieTitle=[ud.ImageFile(1:end-4) '_DispVect'];
      set(ud.h.MovieString,'String',['Curr. movie: ' ud.MovieTitle]);
      set(DemoFig,'UserData',ud);
      %SaveMovie;  % allow user to save movie
   else
      delete(12)
   end

case 2    %x-displacements
   figure(13)
   set(13,'Name','x-
Displacement','NumberTitle','off','CloseRequestFcn','SMPavi("DisplayAbort");','Userdata',0)
   ud.Movie=[];
   % find extreme value for displacement
   minD=realmax;
   maxD=-realmax;
   for n=1:Nframes
      if max(max(ud.Results{n}(:,:,3)))>maxD
         maxD=max(max(ud.Results{n}(:,:,3)))*ud.Scale;
      end
      if min(min(ud.Results{n}(:,:,3)))<minD
         minD=min(min(ud.Results{n}(:,:,3)))*ud.Scale;
```

```
    end
  end
  if ~isnan(ud.CbarMax)      % apply manual colorbar limits
    maxD=ud.CbarMax;
  end
  if ~isnan(ud.CbarMin)
    minD=ud.CbarMin;
  end
  %disp('Animation: hit key to continue ...')
  for n=1:Nframes
    stop=get(13,'userdata');
    if stop
      break
    end
    if gcf~=13
      figure(13)
      % set(13,'Name','x-Displacement','NumberTitle','off')
    end
    x=ud.Results{n}(:,:,1);                    % Result vector x-positon
    y=ud.Results{n}(:,:,2);                    % result vector y-position
    dx=ud.Results{n}(:,:,3)*ud.Scale;                  % x-displacement
    dy=ud.Results{n}(:,:,4)*ud.Scale;                  % y-displacement
    XcorrMax=ud.Results{n}(:,:,5);          % cross-correlation maxima
    ErrorIndex=ud.Results{n}(:,:,6);% error index
    % 2-6-02: use filters for result display
    meanInt=ud.Results{n}(:,:,7);   % mean intensity of each kernel
    KernelStd=ud.Results{n}(:,:,8); % kernel standard deviation
    avgMean=mean2(meanInt);
    ErrorIndex(meanInt<avgMean*ud.MeanIntThreshold)=6;          % apply intensity filter
    ErrorIndex(XcorrMax<ud.XcorrMaxThreshold)=5;       % apply x-corr-maxima filter
    ErrorIndex(KernelStd<ud.StdMin)=4;              % apply std-filter

    dx(ErrorIndex>0)=0; % neglect results with error
    dy(ErrorIndex>0)=0; % neglect results with error
    if Interpolation==1    % no interpolation
      imagesc([ud.Results{n}(1,1,1) ud.Results{n}(1,end,1)],[ud.Results{n}(1,1,2)
ud.Results{n}(end,1,2)],dx,[minD maxD+eps])
      axis image
      title('x-Displacement')
      % possibly chance axis
    else
      switch Interpolation
      case 2
        xDispInt=interp2(x,y,dx,xi,yi,'linear');       %interpolated results
      case 3
        xDispInt=interp2(x,y,dx,xi,yi,'nearest');      %interpolated results
      otherwise
        xDispInt=interp2(x,y,dx,xi,yi,'cubic');        %interpolated results
      end
      imagesc([ud.Results{n}(1,1,1) ud.Results{n}(1,end,1)],[ud.Results{n}(1,1,2)
ud.Results{n}(end,1,2)],xDispInt,[minD maxD+eps])
      axis image
      title('x-Displacement (interpolated)')
    end
    colorbar
    xlabel(['frame: ' num2str(n) ' / ' num2str(Nframes)])
```

```
      M(n)=getframe(13);
      if strcmp(lower(ud.Pause),'yes') & n<Nframes       % pause if ud.Pause is activated and we are not at
last frame
         pause
      end
   end
   if n==Nframes
      set(13,'CloseRequestFcn','delete(gcbf)')
      disp(' ... done! Movie contained in ud.Movie')
      ud.Movie=M;
      ud.MovieTitle=[ud.ImageFile(1:end-4) '_Xdisp'];
      set(ud.h.MovieString,'String',['Curr. movie: ' ud.MovieTitle]);
      set(DemoFig,'UserData',ud);
      %SaveMovie;  % allow user to save movie
   else
      delete(13)
   end
case 3    % y-displacements
   figure(14)
   set(14,'Name','y-
Displacement','NumberTitle','off','CloseRequestFcn','SMPavi("DisplayAbort");','Userdata',0)
   ud.Movie=[];
   % find extreme value for displacement
   minD=realmax;
   maxD=-realmax;
   for n=1:Nframes
      if max(max(ud.Results{n}(:,:,4)))>maxD
         maxD=max(max(ud.Results{n}(:,:,4)))*ud.Scale;
      end
      if min(min(ud.Results{n}(:,:,4)))<minD
         minD=min(min(ud.Results{n}(:,:,4)))*ud.Scale;
      end
   end
   if ~isnan(ud.CbarMax)      % apply manual colorbar limits
      maxD=ud.CbarMax;
   end
   if ~isnan(ud.CbarMin)
      minD=ud.CbarMin;
   end
   for n=1:Nframes
      stop=get(14,'userdata');
      if stop
         break
      end
      if gcf~=14
         figure(14)
         %set(14,'Name','y-Displacement','NumberTitle','off')
      end
      x=ud.Results{n}(:,:,1);                              % Result vector x-positon
      y=ud.Results{n}(:,:,2);                              % result vector y-position
      dx=ud.Results{n}(:,:,3)*ud.Scale;                          % x-displacement
      dy=ud.Results{n}(:,:,4)*ud.Scale;                          % y-displacement
      XcorrMax=ud.Results{n}(:,:,5);          % cross-correlation maxima
      ErrorIndex=ud.Results{n}(:,:,6);% error index
      % 2-6-02: use filters for result display
      meanInt=ud.Results{n}(:,:,7);   % mean intensity of each kernel
```

```
      KernelStd=ud.Results{n}(:,:,8); % kernel standard deviation
      avgMean=mean2(meanInt);
      ErrorIndex(meanInt<avgMean*ud.MeanIntThreshold)=6;                % apply intensity filter
      ErrorIndex(XcorrMax<ud.XcorrMaxThreshold)=5;      % apply x-corr-maxima filter
      ErrorIndex(KernelStd<ud.StdMin)=4;                % apply std-filter

      dx(ErrorIndex>0)=0; % neglect results with error
      dy(ErrorIndex>0)=0; % neglect results with error
      if Interpolation==1    % no interpolation
        imagesc([ud.Results{n}(1,1,1) ud.Results{n}(1,end,1)],[ud.Results{n}(1,1,2)
ud.Results{n}(end,1,2)],dy,[minD maxD+eps])
        axis image
        title('y-Displacement')
      else
        switch Interpolation
        case 2
          yDispInt=interp2(x,y,dy,xi,yi,'linear');        %interpolated results
        case 3
          yDispInt=interp2(x,y,dy,xi,yi,'nearest');       %interpolated results
        otherwise
          yDispInt=interp2(x,y,dy,xi,yi,'cubic');         %interpolated results
        end
        imagesc([ud.Results{n}(1,1,1) ud.Results{n}(1,end,1)],[ud.Results{n}(1,1,2)
ud.Results{n}(end,1,2)],yDispInt,[minD maxD+eps])
        axis image
        title('y-Displacement (interpolated)')
      end
      colorbar
      xlabel(['frame: ' num2str(n) ' / ' num2str(Nframes)])
      M(n)=getframe(14);
      if strcmp(lower(ud.Pause),'yes') & n<Nframes       % pause if ud.Pause is activated and we are not at
last frame
        pause
      end
    end
    if n==Nframes
      set(14,'CloseRequestFcn','delete(gcbf)')
      disp(' ... done! Movie contained in ud.Movie')
      ud.Movie=M;
      ud.MovieTitle=[ud.ImageFile(1:end-4) '_Ydisp'];
      set(ud.h.MovieString,'String',['Curr. movie: ' ud.MovieTitle]);
      set(DemoFig,'UserData',ud);
      %SaveMovie;  % allow user to save movie
    else
      delete(14)
    end
  case 4    % absolute displacements
    figure(15)
    set(15,'Name','Absolute
Displacement','NumberTitle','off','CloseRequestFcn','SMPavi("DisplayAbort");','Userdata',0)
    ud.Movie=[];
    % find extreme value for displacement
    minD=realmax;
    maxD=-realmax;
    for n=1:Nframes
      absD=sqrt(ud.Results{n}(:,:,3).^2+ud.Results{n}(:,:,4).^2)*ud.Scale;
```

```
    if max(max(absD))>maxD
      maxD=max(max(absD));
    end
    if min(min(absD))<minD
      minD=min(min(absD));
    end
  end
  if ~isnan(ud.CbarMax)      % apply manual colorbar limits
    maxD=ud.CbarMax;
  end
  if ~isnan(ud.CbarMin)
    minD=ud.CbarMin;
  end
  for n=1:Nframes
    stop=get(15,'userdata');
    if stop
      break
    end
    if gcf~=15
      figure(15)
      % set(15,'Name','Absolute Displacement','NumberTitle','off')
    end
    x=ud.Results{n}(:,:,1);                    % Result vector x-positon
    y=ud.Results{n}(:,:,2);                    % result vector y-position
    dx=ud.Results{n}(:,:,3)*ud.Scale;                  % x-displacement
    dy=ud.Results{n}(:,:,4)*ud.Scale;                  % y-displacement
    XcorrMax=ud.Results{n}(:,:,5);          % cross-correlation maxima
    ErrorIndex=ud.Results{n}(:,:,6);% error index
    % 2-6-02: use filters for result display
    meanInt=ud.Results{n}(:,:,7);  % mean intensity of each kernel
    KernelStd=ud.Results{n}(:,:,8); % kernel standard deviation
    avgMean=mean2(meanInt);
    ErrorIndex(meanInt<avgMean*ud.MeanIntThreshold)=6;          % apply intensity filter
    ErrorIndex(XcorrMax<ud.XcorrMaxThreshold)=5;      % apply x-corr-maxima filter
    ErrorIndex(KernelStd<ud.StdMin)=4;          % apply std-filter

    dx(ErrorIndex>0)=0; % neglect results with error
    dy(ErrorIndex>0)=0; % neglect results with error
    absDisp=sqrt(dx.^2+dy.^2);
    if Interpolation==1    % no interpolation
      imagesc([ud.Results{n}(1,1,1) ud.Results{n}(1,end,1)],[ud.Results{n}(1,1,2)
ud.Results{n}(end,1,2)],absDisp,[minD maxD+eps])
      axis image
      title('Absolute Displacement')
    else
      switch Interpolation
      case 2
        aDispInt=interp2(x,y,absDisp,xi,yi,'linear'); %interpolated results
      case 3
        aDispInt=interp2(x,y,absDisp,xi,yi,'nearest');        %interpolated results
      otherwise
        aDispInt=interp2(x,y,absDisp,xi,yi,'cubic'); %interpolated results
      end
      imagesc([ud.Results{n}(1,1,1) ud.Results{n}(1,end,1)],[ud.Results{n}(1,1,2)
ud.Results{n}(end,1,2)],aDispInt,[minD maxD+eps]), axis image
      title('Absolute Displacement (interpolated)')
```

```
      end
      colorbar
      xlabel(['frame: ' num2str(n) ' / ' num2str(Nframes)])
      M(n)=getframe(15);
      if strcmp(lower(ud.Pause),'yes') & n<Nframes       % pause if ud.Pause is activated and we are not at
last frame
         pause
      end
   end
   if n==Nframes
      set(15,'CloseRequestFcn','delete(gcbf)')
      disp(' ... done! Movie contained in ud.Movie')
      ud.Movie=M;
      ud.MovieTitle=[ud.ImageFile(1:end-4) '_absDisp'];
      set(ud.h.MovieString,'String',['Curr. movie: ' ud.MovieTitle]);
      set(DemoFig,'UserData',ud);
      %SaveMovie;  % allow user to save movie
   else
      delete(15)
   end
case 5    % cross-correlation maximum
   figure(16)
   set(16,'Name','Cross-Correlation
Maxima','NumberTitle','off','CloseRequestFcn','SMPavi("DisplayAbort");','Userdata',0)
   ud.Movie=[];
   %disp('Animation: hit key to continue ...')
   for n=1:Nframes
      stop=get(16,'userdata');
      if stop
         break
      end
      if gcf~=16
         figure(16)
         %set(16,'Name','Cross-Correlation Maxima','NumberTitle','off')
      end
      XcorrMax=ud.Results{n}(:,:,5);          % cross-correlation maxima
      imagesc([ud.Results{n}(1,1,1) ud.Results{n}(1,end,1)],[ud.Results{n}(1,1,2)
ud.Results{n}(end,1,2)],XcorrMax,[0 1])
      axis image
      title('Cross-Correlation Maxima')
      colorbar
      xlabel(['frame: ' num2str(n) ' / ' num2str(Nframes)])
      M(n)=getframe(16);
      if strcmp(lower(ud.Pause),'yes') & n<Nframes       % pause if ud.Pause is activated and we are not at
last frame
         pause
      end
   end
   if n==Nframes
      set(16,'CloseRequestFcn','delete(gcbf)')
      disp(' ... done! Movie contained in ud.Movie')
      ud.Movie=M;
      ud.MovieTitle=[ud.ImageFile(1:end-4) 'XcorrMax'];
      set(ud.h.MovieString,'String',['Curr. movie: ' ud.MovieTitle])
      set(DemoFig,'UserData',ud);
      %SaveMovie;  % allow user to save movie
```

```
  else
    delete(16)
  end
case 6   % error-plot
  figure(17)
  set(17,'Name','Cross-Correlation
Error','NumberTitle','off','CloseRequestFcn','SMPavi("DisplayAbort");','Userdata',0)
  ud.Movie=[];
  % find extreme value for displacement
  % disp('Animation: hit key to continue ...')
  for n=1:Nframes
    stop=get(17,'userdata');
    if stop
      break
    end
    if gcf~=17
      figure(17)
      %  set(17,'Name','Cross-Correlation Error','NumberTitle','off')
    end
    XcorrMax=ud.Results{n}(:,:,5);          % cross-correlation maxima
    ErrorIndex=ud.Results{n}(:,:,6);% error index
    % 2-6-02: use filters for result display
    meanInt=ud.Results{n}(:,:,7);  % mean intensity of each kernel
    KernelStd=ud.Results{n}(:,:,8); % kernel standard deviation
    avgMean=mean2(meanInt);
    ErrorIndex(meanInt<avgMean*ud.MeanIntThreshold)=6;                % apply intensity filter
    ErrorIndex(XcorrMax<ud.XcorrMaxThreshold)=5;      % apply x-corr-maxima filter
    ErrorIndex(KernelStd<ud.StdMin)=4;              % apply std-filter

    imagesc([ud.Results{n}(1,1,1) ud.Results{n}(1,end,1)],[ud.Results{n}(1,1,2)
ud.Results{n}(end,1,2)],ErrorIndex,[0 6])
    axis image
    title('Cross-Correlation Error')
    colorbar
    xlabel(['frame: ' num2str(n) ' / ' num2str(Nframes)])
    M(n)=getframe(17);
    if strcmp(lower(ud.Pause),'yes') & n<Nframes      % pause if ud.Pause is activated and we are not at
last frame
      pause
    end
  end
  if n==Nframes
    set(17,'CloseRequestFcn','delete(gcbf)')
    disp(' ... done! Movie contained in ud.Movie')
    ud.Movie=M;
    ud.MovieTitle=[ud.ImageFile(1:end-4) '_XcorrErr'];
    set(ud.h.MovieString,'String',['Curr. movie: ' ud.MovieTitle]);
    set(DemoFig,'UserData',ud);
    %SaveMovie;  % allow user to save movie
  else
    delete(17)
  end
case 7   % kernel standard deviations maximum
  figure(18)
  set(18,'Name','Kernel Standard
Deviations','NumberTitle','off','CloseRequestFcn','SMPavi("DisplayAbort");','Userdata',0)
```

```
   ud.Movie=[];
   %disp('Animation: hit key to continue ...')
   for n=1:Nframes
      stop=get(18,'userdata');
      if stop
         break
      end
      if gcf~=18
         figure(18)
         %set(18,'Name','Kernel Standard Deviations','NumberTitle','off')
      end
      KernelStd=ud.Results{n}(:,:,8);          % cross-correlation maxima
      imagesc([ud.Results{n}(1,1,1) ud.Results{n}(1,end,1)],[ud.Results{n}(1,1,2)
ud.Results{n}(end,1,2)],KernelStd,[0 30])
      axis image
      title('Kernel Standard Deviations')
      colorbar
      xlabel(['frame: ' num2str(n) ' / ' num2str(Nframes)])
      M(n)=getframe(18);
      if strcmp(lower(ud.Pause),'yes') & n<Nframes      % pause if ud.Pause is activated and we are not at
last frame
         pause
      end
   end
   if n==Nframes
      set(18,'CloseRequestFcn','delete(gcbf)')
      disp(' ... done! Movie contained in ud.Movie')
      ud.Movie=M;
      ud.MovieTitle=[ud.ImageFile(1:end-4) 'XcorrMax'];
      set(ud.h.MovieString,'String',['Curr. movie: ' ud.MovieTitle])
      set(DemoFig,'UserData',ud);
      %SaveMovie;  % allow user to save movie
   else
      delete(18)
   end
case 8    % image difference
   figure(19)
   set(19,'Name','Image
Difference','NumberTitle','off','CloseRequestFcn','SMPavi("DisplayAbort");','Userdata',0)
   ud.Movie=[];
   % find extreme value for displacement
   minD=realmax;
   maxD=-realmax;
   for n=1:Nframes
      file=fullfile(ud.Pathname,ud.ImageFile);
      if ud.RelDisp==1;
         avi=aviread(file,n);
      else
         avi=aviread(file,1);
      end
      im1=avi.cdata;
      avi=aviread(file,n+1);
      im2=avi.cdata; clear avi;
      %D=imsubtract(im2,im1);
      D=double(im2)-double(im1);
      if max(D(:))>maxD
```

```
        maxD=max(D(:));
      end
      if min(D(:))<minD
        minD=min(D(:));
      end
    end
    if ~isnan(ud.CbarMax)      % apply manual colorbar limits
      maxD=ud.CbarMax;
    end
    if ~isnan(ud.CbarMin)
      minD=ud.CbarMin;
    end
    clear im1 im2
    for n=1:Nframes
      stop=get(19,'userdata');
      if stop
        break
      end
      if gcf~=19
        figure(19)
        %  set(19,'Name','Image Difference','NumberTitle','off')
      end
      file=fullfile(ud.Pathname,ud.ImageFile);
       if ud.RelDisp==1;
        avi=aviread(file,n);
      else
        avi=aviread(file,1);
      end
      im1=avi.cdata;
      avi=aviread(file,n+1);
      im2=avi.cdata;
      ImgDiff=double(im2)-double(im1);
      imagesc(ImgDiff,[minD maxD]); axis image
      title('Image Difference: frame(n+1) - frame(n)')
      colorbar
      xlabel(['frame: ' num2str(n) ' / ' num2str(Nframes)])
      M(n)=getframe(19);
      if strcmp(lower(ud.Pause),'yes') & n<Nframes       % pause if ud.Pause is activated and we are not at
last frame
        pause
      end
    end
    if n==Nframes
      set(19,'CloseRequestFcn','delete(gcbf)')
      disp(' ... done! Movie contained in ud.Movie')
      ud.Movie=M;
      ud.MovieTitle=[ud.ImageFile(1:end-4) '_ImgDiff'];
      set(ud.h.MovieString,'String',['Curr. movie: ' ud.MovieTitle]);
      set(DemoFig,'UserData',ud);
     % SaveMovie;  % allow user to save movie
    else
      delete(19)
    end
otherwise
    disp('invalid selection')
end
```

```
%%
%% Sub-function SaveResults
function SaveResults
DemoFig = gcbf;
ud = get(DemoFig, 'UserData');
oldDir=cd;                    % remember old directory
cd(ud.Pathname);
initfile=[ud.ImageFile(1:end-4) '_SMPavi_results.mat'];
[filename, pathname]=uiputfile(initfile,'Save Workspace');
cd(oldDir);
if filename
   [path, name, ext, ver]=fileparts(filename);
   file=fullfile(pathname,filename);
   if strcmp(lower(ext),'.csv')        %export results as csv file
      answer = inputdlg(['Which step do you want to save (1 - ' num2str(size(ud.Results,2)) '): '], 'Save
results as CSV-file', 1);
      if ~isempty(answer)
         n = median([1 round(str2num(answer{1})) size(ud.Results,2)]);
         % prepare data structure: results will be saved in columns:
         % column 1: x-center, 2: y-center, 3: x-displ. 4: y-displc. 5: x-corr max. 6: error index 7: mean
intensity 8: stdv
         col1=ud.Results{n}(:,:,1);
         col2=ud.Results{n}(:,:,2);
         col3=ud.Results{n}(:,:,3);
         col4=ud.Results{n}(:,:,4);
         col5=ud.Results{n}(:,:,5);
         col6=ud.Results{n}(:,:,6);
         col7=ud.Results{n}(:,:,7);
         col8=ud.Results{n}(:,:,8);
         storeData=[col1(:) col2(:) col3(:) col4(:) col5(:) col6(:) col7(:) col8(:)];
         csvwrite(file,storeData)
         %disp('saved as csv')
      end
   else
      % prepare filestructure
      data=ud; data.FigureHandle=[]; data.h=[]; data.t=[]; data.Movie=[]; data.MovieTitle=[];
      save(file,'data');
   end
end

%%
%% Sub-function LoadResults
function LoadResults
DemoFig = gcbf;
ud = get(DemoFig, 'UserData');
oldDir=cd;
cd(ud.Pathname);
[filename, pathname]=uigetfile('*.mat','Load Results');
cd(oldDir);
if filename
   data=[];
   file=fullfile(pathname,filename);
   load(file)
   %delete old ROI rectangle
   if ishandle(ud.h.ROIhandle)
```

```
    delete(ud.h.ROIhandle); % ud.h.ROIhandle=[];
  end
  % update Userdata, keep FigureHandle and GUI controls
  data.h=ud.h; data.FigureHandle=ud.FigureHandle;
  ud=data;
  set(DemoFig,'UserData',ud);
  % update enable buttons
  set([ud.h.CrossCorrelation ud.h.SaveResults],'Enable','on');
  if size(ud.Results)
    set(ud.h.DisplayResults,'Enable','on');
  end
  %update user data and images
  ud.CurrentImage=1;
  if size(ud.ROIpoly)       %update ROI polygone
    RectX=ud.ROIpoly(:,1);
    RectY=ud.ROIpoly(:,2);
    ud.h.ROIhandle = line('XData', RectX, ...
      'YData', RectY, ...
      'Parent', ud.h.ImageAxes, ...
      'Clipping', 'on', ...
      'Color', 'r', ...
      'LineStyle', '-', ...
      'EraseMode', 'normal');
  end
  img=ud.ImageData{1};
  minimum=double(min(img(:)));
  maximum=double(max(img(:)));
  ud.MinMax=[minimum maximum];
  set(ud.h.SetMin,'String',num2str(minimum));
  set(ud.h.SetMax,'String',num2str(maximum));
  set(DemoFig,'UserData',ud);
  set(ud.h.ImageImage, 'Cdata', img,...
    'XData',[1 size(img,2)],'YData',[1 size(img,1)]);
  set(ud.h.ImageAxes,'XLim',[1 size(img,2)],'YLim',[1 size(img,1)]);
  set(ud.h.FigureString,'String',['Figure 1: ' ud.ImageFile]);
end

%%
%% Sub-function SaveMovie
function SaveMovie
DemoFig = findobj(allchild(0), 'tag', 'SMPavi Program');        % find handle for main menu
ud = get(DemoFig, 'UserData');
if size(ud.Movie)
  %disp('Saving movie to avi-file')
  %fps=input(['Frames / second (' num2str(ud.infoavi.NumFrames) ' frames total): ? ']);
  prompt={['Frames / second (' num2str(ud.infoavi.NumFrames) ' frames total): ? ']};
  dlgTitle='Saving movie to avi-file';
  def={'2'};
  answer=inputdlg(prompt,dlgTitle,1,def);
  if length(answer)
    fps=str2num(answer{1});
    if isa(fps,'double') & fps>0
      oldDir=cd;                    % remember old directory
      cd(ud.Pathname);
      %[path, name, ext, ver]=fileparts(ud.Filename);
      %initfile=[name '_workspace' '.mat'];
```

```
        initfile=[ud.MovieTitle '.avi'];
        [filename, pathname]=uiputfile(initfile,'Save results as avi-file');
        cd(oldDir);
        if filename
          file=fullfile(pathname,filename);
          movie2avi(ud.Movie,file,'FPS',fps,'COMPRESSION','none');
          disp(['saved avi-file as ' filename])
        end
      end
    end
end

%%
%% Sub-function HelpWindow
function HelpWindow
help_title1='Strain Mapping Program';
help_str1=char('The program uses a cross-correlation peak method to determine the', ...
  'relative displacement between corresponding regions from two images.',...
  'The user loads the images using the <browse> buttons and can threshold',...
  'the image by adjusting the <min> and <max> fields.',...
  'The <histogram> button can aid in determininig the intensity level of', ...
  'the background.',...
  'Once two images have been loaded into the program the <cross-correlation',...
  'button becomes available. Clicking this button leads to a new window in',...
  'which the user can choose the maximal expected displacement, the size of',...
  'the regions that will be correlated and the spacing between regions.',...
  'The region of interest <ROI> button allows to limit the strain mapping',...
  'to a selected region of interest of the image and speeds up the process.');
help_title2='Keyboard Functions';
help_str2=char('Several functions can only be activated using the keyboard.',...
  'Here is a brief overview:',...
  '    <h>   Help menu (this window)',...
  '    <ESC>  Quit program',...
  '    <p>   edit Parameters (scale, result display options)',...
  '    <f>   edit Filter for result display (requirements for valid vectors)',...
  '    <s>   Save current movie',...
  '    <e>   Erase current movie from memory');
help_title3='Display Result';
help_str3=char('Once the cross-correlation algorithm has finished the results can be',...
  'displayed by selecting the desired display with the pull-out menu and',...
  'pressing the <Display Reusults> button.',...
  'A second pull out menu contains different options for the selected result mode:',...
  'It can be used to choose different modes of interpolation or to scale',...
  'displacement vectors.',...
  'The available result modes are:',...
  'disp. vectors:    relative displacement vectors)',...
  'x-displacement:   magnitude of displacement in x-direction',...
  'y-displacement:   magnitude of displacement in y-direction',...
  'abs. disp.:       absolute displacement',...
  'x-corr max.       value of cross-correlation peak (between 0 and 1)',...
  'x-corr. error:    error indicator:',...
  '              0 - no error, valid result',...
  '              1 - comp. displacement too large (only for subpixel)',...
  '              2 - x-correlation peak at matrix edge (only for subpixel)',...
  '              3 - no unique x-correlation peak',...
  '              4 - kernel standard deviation < minimum standard deviation (default: 10)',...
```

```
  '              5 - x-correlation peak < minimum x-correlation peak (default: 0.5)',...
  '              6 - intensity average below threshold',...
  'image difference:  image(1) - image(2)');
help_title4='Save Results';
help_str4=char('Default option saves all the results, together with the image',...
  'information, cross-correlation settings and region of interest.',...
  'Analysis that has been saved using this option can be resumed and modified',...
  'at a later time.',...
  'If the user chooses .csv as the file extension, then the numeric results will',...
  'be exported as a csv - comma separated values- file.',...
  'The file structure is:',...
  'column 1: x-center of correlation window',...
  '      2: y-center',...
  '      3: x-displacement',...
  '      4: y-displacement',...
  '      5: x-correlation maximum (peak value)',...
  '      6: error index (see previous section)',...
  '      7: kernel intensity',...
  '      8: kernel standard deviation');
help_str={help_title1 help_str1; help_title2 help_str2; help_title3 help_str3; help_title4 help_str4};
helpwin(help_str)

%%
%% Sub-function EditFilter
function EditFilter
%DemoFig = gcbf;
DemoFig = findobj(allchild(0), 'tag', 'SMPavi Program');        % find handle for main menu
ud = get(DemoFig, 'UserData');
% input dialog for XcorrMaxThreshold and minimum for standard deviation requirement for valid vector
%prompt={'Enter minimum for cross-correlation maxium (0-1.0)','Enter minimum for standard deviation
within kernel (0-255)'};
%def={num2str(ud.XcorrMaxThreshold), num2str(ud.StdMin)};
prompt={'Enter minimum for cross-correlation maxium (0-1.0)','Enter minimum for standard deviation
within kernel (0-255)',...
     'Enter minimum for kernel intensity as factor of mean intensity (0-2.0)','Process every n-th frame (1-
100)'};
def={num2str(ud.XcorrMaxThreshold), num2str(ud.StdMin), num2str(ud.MeanIntThreshold),
num2str(ud.skipFrame)};
dlgTitle='Criteria for valid displacement vector';
lineNo=1;
answer=inputdlg(prompt,dlgTitle,lineNo,def);
if length(answer)
  ud.XcorrMaxThreshold=median([0 str2num(answer{1}) 1]);   % make sure that value is between 0 and
1
  ud.StdMin=median([0 str2num(answer{2}) 255]); % make sure value is between 0 and 255
  ud.MeanIntThreshold=median([0 str2num(answer{3}) 2]);
  ud.skipFrame=median([1 str2num(answer{4}) 100]);
  set(DemoFig,'UserData',ud);
  if ishandle(ud.t.skipFrame)
    set(ud.t.skipFrame, 'String',['process every ' num2str(ud.skipFrame) ' frame(s)'])
  end

  %set(ud.t.minXcorr, 'String',['min. x-correlation value: ' num2str(ud.XcorrMaxThreshold)]);
  %set(ud.t.minStd, 'String',['min. standard deviation: ' num2str(ud.StdMin)]);
end
```

```
%%
%% Sub-function EditParameter
function EditParameter
DemoFig = findobj(allchild(0), 'tag', 'SMPavi Program');        % find handle for main menu
ud = get(DemoFig, 'UserData');
% input dialog for program parameters
prompt={'Enter magnification scale (micron/pixel)','Pause between frames when displaying results
(yes/no)',...
    'Relative or absolute displacements (rel/abs)','Colorbar max (<NaN> for auto)','Colorbar min (<NaN>
for auto)'};
txt=char('abs','rel');
def={num2str(ud.Scale), ud.Pause, txt(ud.RelDisp+1,:), num2str(ud.CbarMax), num2str(ud.CbarMin)};
dlgTitle='Edit Program Parameter';
lineNo=1;
answer=inputdlg(prompt,dlgTitle,lineNo,def);
if length(answer)
  ud.Scale=median([0 str2num(answer{1}) 10]);   % make sure that value is between 0 and 10
  ud.Pause=answer{2};
  switch lower(answer{3})
  case 'rel'
    ud.RelDisp=1;
  otherwise
    ud.RelDisp=0;
  end
  ud.CbarMax=str2num(answer{4});
  if ~size(ud.CbarMax)
    ud.CbarMax=NaN;
  end
  ud.CbarMin=str2num(answer{5});
  if ~size(ud.CbarMin)
    ud.CbarMin=NaN;
  end

  set(DemoFig,'UserData',ud);
end

%%
%% Sub-function Erase Movie
function EraseMovie
DemoFig = findobj(allchild(0), 'tag', 'SMPavi Program');        % find handle for main menu
ud = get(DemoFig, 'UserData');
set(ud.h.MovieString,'String',['Current movie: ' ud.MovieTitle]);
ud.Movie=[];
ud.MovieTitle=[];
set(DemoFig,'UserData',ud);

%%
%% Sub-function Display Abort
function DisplayAbort
set(gcbf,'userdata',1);
```

Single sarcomere tracking progams "SSTPget", "SSTPdo", and "SSTPplot"

"SSTPget"

```
% single sarcomere tracking program
% this module is for loading avis and creating the line-intensity data
% Jan Lammerding
% 9/25/02
% 10/7/02
clear
oldCD=cd;
Factor=10;  % interpolation factor
na=2;      % number of lines to use for averaging
filename='default';
% Pre-processing: Getting Intensity along line from avi
while filename
   [filename, pathname]=uigetfile('*.avi','Select myocyte avi-file');
   if filename
      cd(pathname);
      file=fullfile(pathname,filename);
      avi=aviread(file,1);
      infoavi=aviinfo(file);
      % update Userdata
      img=avi.cdata;
      clear avi
      figure(1), set(1,'Name',filename,'NumberTitle','off');
      imagesc(img), axis image, colormap gray
      %title('zoom in, then hit key')
      %zoom(1,'on')
      %pause
      %zoom(1,'off')
      title('Mark bead center')
      [Bx, By]=ginput(1);
      Bx=round(Bx(1)); By=round(By(1));
      pause(0.1);
      nLine=0;
      cont=1;
      while cont
         nLine=nLine+1;
         title(['select sarcomere ' num2str(nLine) ': left button - start  right button - end']);
         [x,y]=getline(1);
         x=round(x(1:2)); y=round(y(1:2));
         title([filename])
         l1(nLine)=line(x,y); set(l1(nLine),'color','r')
         set(l1(nLine),'LineWidth',2)
         h1(nLine)=text(x(2),y(2),['L ' num2str(nLine)]); % label line
         %[th(nLine), r(nLine)]=cart2pol(x(2)-x(1),y(2)-y(1));
         [th, r]=cart2pol(x(2)-x(1),y(2)-y(1));
         xc(:,nLine)=x(:); yc(:,nLine)=y(:);
         %xc(2,nLine)=xc(1,nLine)+cos(th(nLine))*round(r(nLine)); %rounded end x
         %yc(2,nLine)=yc(1,nLine)+sin(th(nLine))*round(r(nLine)); %rounded end y
         xc(2,nLine)=xc(1,nLine)+cos(th)*round(r); %rounded end x
         yc(2,nLine)=yc(1,nLine)+sin(th)*round(r); %rounded end y
         xi0=linspace(xc(1,nLine),xc(2,nLine),round(r)*Factor);
         yi0=linspace(yc(1,nLine),yc(2,nLine),round(r)*Factor);
         %xi0{nLine}=linspace(xc(1,nLine),xc(2,nLine),round(r(nLine))*Factor);
         %yi0{nLine}=linspace(yc(1,nLine),yc(2,nLine),round(r(nLine))*Factor);
         %z0{nLine}=interp2(double(img), xi0{nLine}, yi0{nLine}, 'cubic');

         int=[-na:na];
```

```
        xis{nLine}=[]; yis{nLine}=[];
        figure(1)
        for i=1:(2*na+1)
          xis{nLine}(i,:)=xi0+int(i)*sin(th);
          yis{nLine}(i,:)=yi0-int(i)*cos(th);
          %xis{nLine}(i,:)=xi0{nLine}+int(i)*sin(th(nLine));
          %yis{nLine}(i,:)=yi0{nLine}-int(i)*cos(th(nLine));
          l3(nLine,i)=line(xis{nLine}(i, [1, length(xi0)]),yis{nLine}(i, [1, length(yi0)]));
set(l3(nLine,i),'color','g')
        end
        zis=interp2(double(img), xis{nLine}, yis{nLine}, 'cubic');
        zi=mean(zis);      % use average as 1-D signal
        Xso=[1:length(zi)]/Factor; % co-ordinates on line (pixel)

        figure(2), set(2,'Name','1D-Data','NumberTitle','off');
        plot(Xso,zi)
        title(['Intensity along line ' num2str(nLine)]);
        xlabel('pixel'); ylabel('intensity')

        button=questdlg('Accept sarcomere boundaries ?','Sarcomere Tracking Program',...
          'Accept','Edit','Discard and Exit','Accept');
        if strcmp(button,'Accept')
          %cont=0;
        else
          delete(l3(nLine,:))
          delete(l1(nLine))
          delete(h1(nLine))
          xc(:,nLine)=[];
          yc(:,nLine)=[];
          nLine=nLine-1;
          figure(2), clf
          if strcmp(button,'Discard and Exit')
            cont=0;
          end
          %
        end
      end
      if nLine          % in case we have some sarcomeres
        % capture line information for entire avi-file
        Nframes=infoavi.NumFrames;
        mag=zeros(1,Nframes);  % information containing magnetic field strength
        h=waitbar(0,'Retrieving Data ...');
        Lintensity=[];
        for n=1:Nframes
          % read current frame from avi-file
          avi=aviread(file,n);
          mag(1,n)=avi.cdata(1,1);
          for i=1:nLine
            %Lintensity{i}=zeros(Nframes,size(xis{i},2));  % intensity along line for all frames
            % get line intensity
            zis=interp2(double(avi.cdata), xis{i}, yis{i}, 'cubic');
            zi=mean(zis);      % use average as 1-D signal
            Lintensity{i}(n,:)=zi;
          end
          waitbar(n/Nframes,h);
        end
```

```
        clear avi
        close(h)

        %% Store Results
        clear Data
        data.factor=Factor;          % interpolation factor
        data.file=filename;          % file name
        data.img=img;                % first frame
        data.intensity=Lintensity;    % intensity data along line
        data.mag=mag;                % magnetic field data
        data.na=na;                  % number of lines to average
        data.xc=xc;                  % line position x
        data.yc=yc;                  % line position y
        data.bx=Bx;                  % bead position x
        data.by=By;                  % bead position y
        % save data
        [sfilename, spathname]=uiputfile([filename(1:end-4) '_SSTdata.mat'],'Save data for future
processing');
        if sfilename
          [path, fname, ext, vers]=fileparts(sfilename);
          file=fullfile(spathname,[fname '.mat']);
          save(file,'data')
          disp(['Displacement data save as ' fname '.mat'])
        end
      end
    end
end
for i=1:2   % close windows
    if ishandle(i)
      close(i)
    end
end
cd(oldCD)
```

"SSTdo"

```
% single sarcomere tracking program
% based on SST4
% this module is for loading data and analyzing sarcomere positions
% Jan Lammerding
% 9/25/02
% 10/01/02:
% included output (two options, CSV or XLS)
% 10/03/02:
% added initial sarcomere position relative to bead
% 10/07/02
% work with SSTPget, i.e. many line-data sets per cell
% remove FFT calculation
% simplify save function (don't export data, only save workspace
% to do:
% incorporate fminJanfunction in case of function calls

clear
data=[];
CMresults=[];
% Load intensity data
[filename, pathname]=uigetfile('*.mat','Select file with SST data');
```

```
if ~filename    % in case no file was selected
   return
end
file=fullfile(pathname,filename);
load(file)
if ~size(data)
   return
end
% Program Parameters    (make some of this interactive)
Nfft=1024;                    % power of FFT analysis
Factor=data.factor;           % interpolation factor
na=data.na;                   % number of lines to use for averaging
Lintensity=data.intensity;    % intensity course along line
mag=data.mag;                 % magnetic field
AVIfile=data.file;            % avi-file name
Nframes=size(Lintensity{1},1);    % number of frames
Nline=size(Lintensity,2);     % number of lines

figure(1), set(1,'Name',filename,'NumberTitle','off'); set(1,'Position',[50 700 300 250]);
imagesc(data.img), axis image, colormap gray
title([data.file]), xlabel('x (pixel)'), ylabel('y (pixel)')
xc=data.xc; yc=data.yc;
bx=data.bx; by=data.by;
b=line([bx-3 bx+3 bx bx bx], [by by by by-3 by+3],'color','r','linewidth',2);    % mark bead

for nLine=1:Nline
   zi=Lintensity{nLine}(1,:);          % intensity at t=0
   Xso=[1:length(zi)]/Factor;      % co-ordinates on line
   [th,r]=cart2pol(xc(2,nLine)-xc(1,nLine),(yc(2,nLine)-yc(1,nLine)));
   thLine(nLine)=th;
   xi0=linspace(xc(1,nLine),xc(2,nLine),round(r)*Factor);
   yi0=linspace(yc(1,nLine),yc(2,nLine),round(r)*Factor);
   int=[-na:na];
   xis{nLine}=[]; yis{nLine}=[];
   figure(1)
   for i=1:(2*na+1)
      xis{nLine}(i,:)=xi0+int(i)*sin(th);
      yis{nLine}(i,:)=yi0-int(i)*cos(th);
      l3=line(xis{nLine}(i, [1, length(xi0)]),yis{nLine}(i, [1, length(yi0)])); set(l3,'color','g')
   end
   l1=line(xc(:,nLine),yc(:,nLine)); set(l1,'color','r')
   set(l1,'LineWidth',2)
   h1=text(xc(2,nLine),yc(2,nLine),['L ' num2str(nLine)]); % label line
end

for nLine=1:Nline;      % do processing for all lines
   zi=Lintensity{nLine}(1,:);          % intensity at t=0
   Xso=[1:length(zi)]/Factor;      % co-ordinates on line

   figure(2), set(2,'Name',['1D-Data Line ' num2str(nLine)],'NumberTitle','off'); set(2,'Position',[50 370 300
250]);
   plot(Xso,zi,'b:')
   axis([Xso(1) Xso(end) 0 255])
   title(['Intensity along line ' num2str(nLine)]);
   xlabel('pixel'); ylabel('intensity')
   Lgd0=[]; Lgd0{1}=['Intensity']; legend(Lgd0)
```

```
figure(3), set(3,'Name',['Time Course I - Line ' num2str(nLine)],'NumberTitle','off'); set(3,'Position',[375
550 550 400]);
imagesc(Lintensity{nLine}); colorbar
title(['Intensity time course line ' num2str(nLine)])
xlabel(['pixel * ' num2str(Factor)]); ylabel('frame')

figure(4), set(4,'Name',['Time Course II - Line ' num2str(nLine)],'NumberTitle','off'); set(4,'Position',[950
700 300 250]);
[frame, pixel]=meshgrid([0:size(Lintensity{nLine},2)-1],[1:size(Lintensity{nLine},1)]);
surf(frame,pixel,Lintensity{nLine},'EdgeColor','none')
axis tight, colorbar, xlabel(['pixel * ' num2str(Factor)]), ylabel('frame'), title('intensity time course')
set(gcf,'Renderer','zbuffer');
view(-15,70)
rotate3d on

% POST-PROCESSING:
% identify sarcomeres:
str=char('left','right');
cont=1;
xSarc=[];
nSarc=0;
h3=[]; h4=[];
colors=char('r','y','g','c','m','k','w','r','y','g','c','m','k','w');
while cont & nSarc<size(colors,1)
   x=zeros(2,Nframes);
   y=[1:Nframes];
   z=zeros(2,Nframes);
   nSarc=nSarc+1;
   for k=1:2
      figure(3)
      title(['Sarcomere ' num2str(nSarc) ' - select minima as sarcomere boundaries: ' str(k,:)]);
      drawnow
      [x0,y0]=ginput(1);
      X0=median([1 round(x0) size(Lintensity{nLine},2)]);
      Y0=median([2 round(y0) Nframes-1]);
      [x(k,Y0), z(k,Y0)] = fminJan(X0,Lintensity{nLine}(Y0,:));
      for i=Y0+1:Nframes
         [x(k,i), z(k,i)] = fminJan(x(k,i-1),Lintensity{nLine}(i,:));
      end
      for i=Y0-1:-1:1
         [x(k,i), z(k,i)] = fminJan(x(k,i+1),Lintensity{nLine}(i,:));
      end
      % process x to get integer values and sort to create left and right boundaries within valid range
      x=round(x);
      x(x>size(Lintensity{nLine},2)) = size(Lintensity{nLine},2);
      x(x<1) = 1;
      %h3(nSarc,k) = line(x(k,:),y); set(h3(nSarc,k),'Color',colors(nSarc,:));
      h3(nSarc,k) = line(x(k,:),y); set(h3(nSarc,k),'color','k');
      figure(4)
      %h4(nSarc,k) = line(x(k,:),y,z(k,:)); set(h4(nSarc,k),'Color',colors(nSarc,:))
      h4(nSarc,k) = line(x(k,:),y,z(k,:)); set(h4(nSarc,k),'color','k')
   end
   x=sort(x);
   figure(3)
```

```
    button=questdlg('Accept sarcomere boundaries ?','Sarcomere Tracking Program','Accept','Accept and
Next','Edit','Accept and Next');
    if strcmp(button,'Accept')
      xSarc{nSarc}=x;
      figure(2), hold on
      x1=[x(1,1):x(2,1)]; y1=Lintensity{nLine}(1,x1);
      plot(x1/Factor, y1, colors(nSarc,:))
      Lgd0{nSarc+1}=['Sarc ' num2str(nSarc)]; legend(Lgd0)
      hold off
      cont=0;
    elseif strcmp(button,'Edit')
      delete(h3(nSarc,[1:2]));
      delete(h4(nSarc,[1:2]));
      nSarc=nSarc-1;
    else      % accept and next
      xSarc{nSarc}=x;
      figure(2), hold on
      x1=[x(1,1):x(2,1)]; y1=Lintensity{nLine}(1,x1);
      plot(x1/Factor, y1, colors(nSarc,:))
      Lgd0{nSarc+1}=['Sarc ' num2str(nSarc)]; legend(Lgd0)
      hold off
    end
  end
  figure(3)
  title('Intensity time course')

  %Sarcomere position based on center of mass
  CMresults{nLine}=zeros(nSarc,Nframes);
  for k=1:nSarc
    x=xSarc{k};
    for n=1:Nframes
      zi=Lintensity{nLine}(n,:);
      X=[x(1,n):x(2,n)];
      Y=zi(X)-min(zi(X));
      CMresults{nLine}(k,n)=sum(X.*Y)/(sum(Y)+eps)/Factor;
    end
  end

  % create legend
  Lgd1=[];
  for i=1:nSarc
    Lgd1{i}=['Sarc ' num2str(i)];
  end

  figure(5), set(5,'Name','Sarcomere Position','NumberTitle','off'); set(5,'Position',[375 220 550 250]);
  plot(y,CMresults{nLine});
  axis([1 Nframes Xso(1) Xso(end)])
  xlabel('frames')
  ylabel('position (pixel)')
  title('Center of mass sarcomere position')
  legend(Lgd1)

  if nSarc>1
    % create legend
    Lgd2=[];
    for i=1:nSarc-1
```

```
        Lgd2{i}=['Length ' num2str(i) ' - ' num2str(i+1)];
      end
      figure(6), set(6,'Name','Sarcomere Length','NumberTitle','off'); set(6,'Position',[950 370 300 250]);
      plot(y,abs(diff(CMresults{nLine})));
      axis([1 Nframes -inf inf]), axis 'auto y'
      xlabel('frames')
      ylabel('length (pixel)')
      title('Sarcomere Length')
      legend(Lgd2)
   else
      figure(6)
      clf
   end

   figure(3)
   imagesc(Lintensity{nLine}); colorbar
   title('Intensity time course')
   xlabel(['pixel * ' num2str(Factor)]); ylabel('frame')
   for i=1:nSarc
      L(i)=line(CMresults{nLine}(i,:)*Factor,y); set(L(i),'Color',colors(i,:));
   end


   % compute position of sarcomere center (1st frame) relative to bead
   thCM=ones(nSarc,1)*thLine(nLine);          % theta
   rCM=CMresults{nLine}(:,1);              % radial position
   [xCM, yCM] = pol2cart(thCM,rCM);   % convert Sarc Centers to cart coord.
   xCM=xCM+xc(1,nLine);
   yCM=yCM+yc(1,nLine);
   [thSarc{nLine}, rSarc{nLine}] = cart2pol(xCM-bx, yCM-by); % convert to polar coordinates relative to
bead

   % plot displacement vectors
   %figure(7), set(7,'Name','Displacement map','NumberTitle','off'); set(7,'Position',[950 40 300 250]);
   %plot(xCM,yCM,'r+')
   %plot(y,FFT_SL,'r',y,fft_SL,'b')
   %xlabel('frames'), ylabel('FFT-sarcomere length (pixel)')
   %axis([1 Nframes -inf inf]), axis 'auto y'
   %title('Sarcomere length')
   %legend('original FFT','center of mass refined')
end

% save results
if size(CMresults)   % make sure that we have results
   [path, AVIfname, ext, vers]=fileparts(AVIfile);
   initfile=[AVIfname '_SSTresults' '.mat'];
   [filename, pathname]=uiputfile(initfile,'Save SST results');
   if filename
      [path, fname, ext, vers]=fileparts(filename);
      file=fullfile(pathname,[fname '.mat']);
      Results=[];
      Results.CMresults=CMresults;   % sarcomere center of mass
      Results.mag=mag;              % magnetic trap data
      Results.rSarc=rSarc;          % Sarcomere position in polar coordinates
      Results.thSarc=thSarc;
      Results.AVIfile=AVIfile;      % name of AVI file
```

```
    Results.thLine=thLine;        % angle of line
    save(file,'Results');
    disp(['Results saved as ' fname '.mat'])
  end
end
```

"SSTPplot"

```
%function [CM, S] = SSTplot
% [CM, S] = function SSTplot
%        CM: centroid information
%        S:  sarcomere information
% single sarcomere tracking analysis program
% to plot results obtained from SSTPdo
% this module is for loading results and creating strain maps
% Jan Lammerding
% requires function files: findpeak, SSTsineanalysis
% 10/08/02
clear
scale=0.3;        % um/pixel
fps=60;           % frame rate
%CM=[];
%SL=[];
filename='default';
cdOld=cd;
Ncell=0;

while filename
  Results=[];
  % Load SSTPdo results
  [filename, pathname]=uigetfile('*.mat','Select file with SST results');
  if filename   % in case no file was selected
    cd(pathname)
    file=fullfile(pathname,filename);
    load(file)
    if ~size(Results)
      disp('invalid file')
    else
      Ncell=Ncell+1;
      % Program Parameters   (make some of this interactive)
      mag{Ncell}=Results.mag;                  % magnetic field
      AVIfile{Ncell}=Results.AVIfile;          % avi-file name
      Nframes{Ncell}=length(mag{Ncell});       % number of frames
      CMresults{Ncell}=Results.CMresults;      % sarcomere centroid positions
      Nline{Ncell}=size(CMresults{Ncell},2);   % number of lines
      rSarc{Ncell}=Results.rSarc;              % sarcomere position in polar coordinates  (radius)
      thSarc{Ncell}=Results.thSarc;            % sarcomere position in polar coordinates  (radius)
      thLine{Ncell}=Results.thLine;            % line angle (in radians)
      clear Results
    end
  end
end

if ~Ncell     % exit if there are no results
  return
end
```

```
for i=1:Ncell;        % compute average magnetic field
  magF(i,:)=mag{i};
end
magF=mean(magF,1);
clear mag

% pool data together and do some pre-processing
% find valid sarcomere length measurements
nSarcL=0;
nSarc=0;
CMinfo=[];
CMcentroid=[];
CMpos=[];
Sinfo=[];
Slength=[];
Spos=[];
for c=1:Ncell
  for i=1:Nline{c}
    CM0=CMresults{c}{i}(:,1);  % sarcomere centroids at t=0 for line i
    for j=1:length(CM0);
      nSarc=nSarc+1;
      CMinfo(nSarc,:) = [c i j thLine{c}(i)];              % centroid info ([ cell line sarcomere line-angle])
      CMcentroid(nSarc,:)= CMresults{c}{i}(j,:);           % centroid position
      CMpos(nSarc,:)=[thSarc{c}{i}(j) rSarc{c}{i}(j)];     % sarcomere position in polar coordinates
[theta radius]
      if j<length(CM0)                                     % test if sarcomere length is valid
        if diff(CM0(j:j+1))*scale<2.5 & diff(CM0(j:j+1))>0  % difference between sarcomere positions
<2.5um => valid sarcomere length
          nSarcL=nSarcL+1;
          Sinfo(nSarcL,:)=[c i j thLine{c}(i)];        % information for valid sarcomere length [cell, line
start sarcomere angle])
          [s1x, s1y] = pol2cart(thSarc{c}{i}(j) , rSarc{c}{i}(j));
          [s2x, s2y] = pol2cart(thSarc{c}{i}(j+1) , rSarc{c}{i}(j+1));
          Spos(nSarcL,:)=[s1x+s2x s1y+s2y]/2;           % sarcomere position [x y]
        end
      end
    end
  end
end
clear CM0;
CMcentroid=sgolayfilt(CMcentroid,3,15,[],2);    % filter results
NsarcL=nSarcL;  % total number of valid sarcomere length
Nsarc=nSarc;    % total number of sarcomeres

% compute sarcomere length
%L=[];
clear Sarc
%Samp=zeros(NsarcL,1);
Slag=zeros(NsarcL,1);
for i=1:NsarcL;
  Ci=Sinfo(i,1);     % indicates cell
  Li=Sinfo(i,2);     % indicates line
  Si=Sinfo(i,3);     % indicates left sarcomere
  Slength(i,:)=abs(diff(CMresults{Ci}{Li}(Si:Si+1,:)));
  [Samp(i,:), Slag(i)] = SSTsineanalysis(Slength(i,:), magF, scale, fps);
```

```
end
Slength=sgolayfilt(Slength,3,15,[],2);    % filter results
clear Li Si CMresults
%CMt0=mean(CMcentroid(:,round([fps/2:fps])),2);
%SLt0=mean(Slength(:,round([fps/2:fps])),2);
CMt0=CMcentroid(:,1);
SLt0=Slength(:,1);
SLt0std=std(Slength(:,round([fps/2:fps])),0,2);

%amp=zeros(Nsarc,1);
lag=zeros(Nsarc,1);
for i=1:Nsarc
  [amp(i,:), lag(i)]=SSTsineanalysis(CMcentroid(i,:),magF,scale,fps);
end
[xi, yi] = meshgrid([-60:60],[-60:60]);

mindL=0;
maxdL=1;
[x, y]=pol2cart(CMpos(:,1),CMpos(:,2));
xS=Spos(:,1);
yS=Spos(:,2);
t=[0:size(CMcentroid,2)-1]/fps;

dispOpt=1;    % use different display options: time course; sarcomere fractional shortening, maximal
displacement, amplitude, phase lag
while dispOpt
  dispOpt=input('Display Option: (1 - disp vectors  2 - disp map  3 - correlation ');
  if ~size(dispOpt)
    dispOpt=0;
  end
  dt=10;
  Vscale=2;

  figure(1)
  plot(x,y,'r+',0,0,'b+',xS,yS,'g+')
  axis ij equal
  axis([-60 60 -60 60])
  xlabel('x (pixel)'), ylabel('y (pixel)')
  if dispOpt>0 & dispOpt<=10
    for i=1:dt:Nframes{1}
      %dL=CMcentroid(:,i)-CMcentroid(:,1);
      dL=CMcentroid(:,i)-CMt0;
      dx=dL.*cos(CMinfo(:,4));
      dy=dL.*sin(CMinfo(:,4));
      %dSL=(Slength(:,i)-Slength(:,1))./Slength(:,1);  % fractional sarcomere shortening
      dSL=(Slength(:,i)-SLt0)./SLt0;  % fractional sarcomere shortening
      dSLx=dSL.*cos(Sinfo(:,4));
      dSLy=dSL.*sin(Sinfo(:,4));
      SL=Slength(:,i);      % sarcomere length
      SLx=SL.*cos(Sinfo(:,4));
      SLy=SL.*sin(Sinfo(:,4));
      figure(2), set(2,'renderer','zbuffer')
      switch dispOpt
      case 1          % Sarcomere Displacement Vectors
        quiver(x*scale, y*scale, dx*scale*Vscale, dy*scale*Vscale, 0)
        axis([-20 20 -20 20])
```

```
        xlabel('x (micron)'), ylabel('y (micron)')
        title(['Sarcomere Displacement - time ' num2str(i) ' / ' num2str(Nframes{1})])
        pause(0.05)
      case 2
        zi_dL=griddata(x,y,dL,xi,yi);
        %imagesc([xi(1,1) xi(1,end)]*scale, [yi(1,1) yi(end,1)]*scale,zi_dL*scale,[mindL maxdL]),
colorbar
        surf(xi*scale,yi*scale,zi_dL*scale,'EdgeColor','none')
        daspect([1 1 0.1])
        axis([-20 20 -20 20 0 1])
        xlabel('x (micron)'), ylabel('y (micron)')
        title(['Sarcomere Displacement - time ' num2str(i) ' / ' num2str(Nframes{1})])
      case 3          % Sarcomere fractional shortening Vectors
        quiver(xS*scale, yS*scale, dSLx*scale*Vscale*100, dSLy*scale*Vscale*100, 0)
        axis([-20 20 -20 20])
        xlabel('x (micron)'), ylabel('y (micron)')
        title(['Sarcomere fractional changes - time ' num2str(i) ' / ' num2str(Nframes{1})])
        pause(0.05)
      case 4
        %zi_dSL=griddata(xS,yS,dSL,xi,yi,'nearest');
        %zi_dSL(isnan(zi_dSL))=-1;
        %imagesc([xi(1,1) xi(1,end)]*scale, [yi(1,1) yi(end,1)]*scale,zi_dSL,[-0.3 0.3]), colorbar
        gt0=dSL>=0;
        clf
        hold on
        stem3(xS(gt0)*scale,yS(gt0)*scale,dSL(gt0),'filled','g')
        stem3(xS(~gt0)*scale,yS(~gt0)*scale,abs(dSL(~gt0)),'filled','r')
        stem3(0,0,0,'b+')
        hold off
        xlabel('x (micron)'), ylabel('y (micron)')
        title(['Sarcomere fractional shortening - time ' num2str(i) ' / ' num2str(Nframes{1})])
        view(-25,70)
        axis([-20 20 -20 20 0 0.5])
        pause
      case 7
        %zi_dSL=griddata(xS,yS,dSL,xi,yi,'nearest');
        %zi_dSL(isnan(zi_dSL))=-1;
        %imagesc([xi(1,1) xi(1,end)]*scale, [yi(1,1) yi(end,1)]*scale,zi_dSL,[-0.3 0.3]), colorbar
        gt0 = abs(dSL)<0.05;
        gtp = dSL>0.05;
        gtn = dSL<-0.05;
        clf
        hold on
        stem3(xS(gt0)*scale,yS(gt0)*scale,dSL(gt0),'filled','c+')
        stem3(xS(gtp)*scale,yS(gtp)*scale,dSL(gtp),'filled','g^')
        stem3(xS(gtn)*scale,yS(gtn)*scale,dSL(gtn),'filled','rv')
        stem3(0,0,0,'b+')
        hold off
        xlabel('x (micron)'), ylabel('y (micron)')
        title(['Sarcomere fractional shortening special- time ' num2str(i) ' / ' num2str(Nframes{1})])
        view(-25,70)
        axis([-20 20 -20 20 0 0.5])
        pause
      case 5          % Sarcomere Length Vectors
        quiver(xS*scale, yS*scale, SLx*scale, SLy*scale, 0)
        axis([-20 20 -20 20])
```

```
        xlabel('x (micron)'), ylabel('y (micron)')
        title(['Sarcomere length - time ' num2str(i) ' / ' num2str(Nframes{1})])
        pause(0.05)
      case 6
        zi_SL=griddata(xS,yS,SL,xi,yi,'nearest');
        imagesc([xi(1,1) xi(1,end)]*scale, [yi(1,1) yi(end,1)]*scale,zi_SL*scale,[1 2.5]), colorbar
        xlabel('x (micron)'), ylabel('y (micron)')
        title(['Sarcomere Length - time ' num2str(i) ' / ' num2str(Nframes{1})])
      end
      drawnow
    end
  end


  if dispOpt==11;
      figure(2)
    %set(3,'renderer','zbuffer')
    %zi_lag=griddata(x,y,lag,xi,yi,'nearest');
    %bar3(zi_lag)
    %imagesc([xi(1,1) xi(1,end)]*scale, [yi(1,1) yi(end,1)]*scale,zi_lag), colorbar
    gt0=lag>=0;
    clf
    hold on
    %stem3(x(gt0)*scale,y(gt0)*scale,lag(gt0),'filled','g')
    %stem3(x(~gt0)*scale,y(~gt0)*scale,abs(lag(~gt0)),'filled','r')
    %stem3(0,0,0,'b+')
    quiver(x*scale,y*scale,lag*20,zeros(size(y)),0)
    plot(0,0,'r+')
    hold off
    xlabel('x (micron)'), ylabel('y (micron)')
    title('Displacement lag')
    figure(3)
    %set(2,'renderer','zbuffer')
    %zi_amp=griddata(x,y,amp,xi,yi,'nearest');
    %bar3(zi_amp)
    %imagesc([xi(1,1) xi(1,end)]*scale, [yi(1,1) yi(end,1)]*scale,zi_amp*scale), colorbar
    clf
    hold on
    quiver(x*scale,y*scale,mean(amp,2)*scale*20,zeros(size(y)),0)
    plot(0,0,'r+')
    %stem3(x*scale,y*scale,mean(amp,2),'filled','g')
    %stem3(0,0,0,'b+')
    hold off
    xlabel('x (micron)'), ylabel('y (micron)')
    title('Displacement amplitude (20x)')
  end

  if dispOpt==12;
    figure(2)
    clf
    hold on
    quiver(xS*scale,yS*scale,Slag*20,zeros(size(yS)),0)
    plot(0,0,'r+')
    hold off
    xlabel('x (micron)'), ylabel('y (micron)')
    title('SL Displacement lag')
```

```
    figure(3)
    clf
    hold on
    quiver(xS*scale,yS*scale,mean(Samp,2)*scale*20,zeros(size(yS)),0)
    plot(0,0,'r+')
    hold off
    xlabel('x (micron)'), ylabel('y (micron)')
    title('Sarcomere length amplitude')
  end
  if dispOpt==13;
    for i=1:size(Samp,2)
      figure(2)
      clf
      hold on
      quiver(x*scale,y*scale,amp(:,i)*scale*20,zeros(size(y)),0)
      plot(0,0,'r+')
      hold off
      xlabel('x (micron)'), ylabel('y (micron)')
      title(['Sarcomere amplitude - period ' num2str(i)])
      pause
    end
  end
  if dispOpt==14;
    for i=1:size(Samp,2)
      figure(2)
      clf
      hold on
      quiver(xS*scale,yS*scale,Samp(:,i)*scale*20,zeros(size(yS)),0)
      plot(0,0,'r+')
      hold off
      xlabel('x (micron)'), ylabel('y (micron)')
      title(['Sarcomere length amplitude - period ' num2str(i)])
      pause
    end
  end
  if dispOpt==20;        % show plots for selected sarcomeres
    figure(2)
    plot(x,y,'r+',0,0,'b+')
    axis ij equal
    axis([-60 60 -60 60])
    xlabel('x (pixel)'), ylabel('y (pixel)')
    title('Select Sarcomere ...')
    [xm, ym]=ginput(1);
    ind=dsearch(x,y,delaunay(x,y),xm,ym)
    hold on
    plot(x(ind),y(ind),'bo'), hold off
    figure(21)
    plot(t,(CMcentroid(ind,:)-CMt0(ind))*scale)
    axis([ 0 t(end) -inf inf])
    xlabel('time (s)'), ylabel('displacement (micron)')
    title(['Sarcomere ' num2str(ind) ': Amplitude ' num2str(amp(ind)) ' um    - phase lag ' num2str(lag(ind))
' sec'])
  end
  if dispOpt==21;        % show plots for selected sarcomere lengths
    figure(2)
    plot(xS,yS,'g+',0,0,'b+')
```

```
      axis ij equal
      axis([-60 60 -60 60])
      xlabel('x (pixel)'), ylabel('y (pixel)')
      title('Select Sarcomere Length...')
      [xm, ym]=ginput(1);
      ind=dsearch(xS,yS,delaunay(xS,yS),xm,ym)
      hold on
      plot(xS(ind),yS(ind),'bo'), hold off
      figure(21)
      plot(t,(Slength(ind,:)-SLt0(ind))*scale)
      axis([ 0 t(end) -inf inf])
      xlabel('time (s)'), ylabel('sarcomere length (micron)')
      title(['Sarcomere Length ' num2str(ind) ': Amplitude ' num2str(Samp(ind)) ' um    - phase lag '
num2str(Slag(ind)) ' sec'])
   end
end
cd(cdOld)
% save results
if size(CMresults)   % make sure that we have results
   [path, AVIfname, ext, vers]=fileparts(AVIfile);
   initfile=[AVIfname '_SSTresults' '.mat'];
   [filename, pathname]=uiputfile(initfile,'Save SST results');
   if filename
      [path, fname, ext, vers]=fileparts(filename);
      file=fullfile(pathname,[fname '.mat']);
      Results=[];
      Results.CMresults=CMresults;   % sarcomere center of mass
      Results.mag=mag;              % magnetic trap data
      Results.rSarc=rSarc;          % Sarcomere position in polar coordinates
      Results.thSarc=thSarc;
      Results.AVIfile=AVIfile;      % name of AVI file
      Results.thLine=thLine;        % angle of line
      save(file,'Results');
      disp(['Results saved as ' fname '.mat'])
   end
end
return
```

Traction force microscopy programs "TFMa" and "TFMb"

"TFMa"

```
% traction force microscopy program
% Jan Lammerding
% 5-9-01
[filename1, pathname]=uigetfile('*.*','Select image 1');
if ~filename1
   return
end
disp(['image 1: ' filename1])
file1=fullfile(pathname,filename1);
im1=imread(file1);
cd(pathname)
[filename2, pathname]=uigetfile('*.*','Select image 2');
if ~filename2
   return
```

```
end
disp(['image 2: ' filename2])
file2=fullfile(pathname,filename2);
im2=imread(file2);

%i1=imread('ft07.tif');
%i2=imread('ft08.tif');
%i3=imread('ft09.tif');
%i4=imread('ft14.tif');
%i5=imread('ft16.tif');
%im1=i1;
%im2=i5;

% adjust image intensities
im1=imadjust(im1,stretchlim(im1,0));
im2=imadjust(im2,stretchlim(im2,0));

figure(1)
imagesc(im1), axis image, colormap gray, title(['image 1: ' filename1])

figure(2)
imagesc(im2), axis image, colormap gray, title(['image 2: ' filename2])

% register images:
h=cpselect(im2,im1);
disp('** Register images **')
disp('Select corresponding beads in both images, then save results and start tfm2b.')
%input('When done, hit return ...')
%dispose(h)
```

"TFMb"

```
% traction force microscopy program
% Jan Lammerding
% 5-9-01
% 2nd part
% store points:
input_pts=input_points;
base_pts=base_points;
clear input_points base_points;

% improve input points using cross-correlation
input_pts_corr=cpcorr(input_pts, base_pts, im2, im1);
% compute transformation based on control point pairs
mytform=cp2tform(input_pts_corr, base_pts, 'linear conformal');
reg_im2=imtransform(im2,mytform,'Xdata',[1 size(im1,2)],'Ydata',[1 size(im1,1)]);
figure(3), imagesc(reg_im2), axis image, colormap gray, title(['image 2: ' filename2 ' - registered'])


pct=99.8;
pct_input=1;
while pct_input
   % automatic bead recognition
   % compute threshold:
   [counts1,x]=imhist(im1);
   [counts2,x]=imhist(im2);
   sc1=cumsum(counts1);   % cumulative histogram
```

```
  sc2=cumsum(counts2);
  thr1=x(min(find(sc1>sum(counts1)*pct/100)));
  thr2=x(min(find(sc2>sum(counts2)*pct/100)));
  % display histogram and threshold
  figure(4), set(4,'Name','Image Histogram with Threshold','NumberTitle','off');
  subplot(2,1,1)
  imhist(im1); title('histogram image 1')
  h1=line([thr1 thr1],[0 max(counts1)]); set(h1,'Color','r');
  subplot(2,1,2)
  imhist(im2); title('histogram image 2')
  h2=line([thr2 thr2],[0 max(counts1)]); set(h2,'Color','r');

    bw1=im1>thr1;        % threshold
  bw2=im2>thr2;
  bead1=bwlabel(bw1,4);  % convert to labeled array
  bead2=bwlabel(bw2,4);
  % identify beads
  stats1 = regionprops(bead1);
  stats2 = regionprops(bead2);
  Areas1 = [stats1.Area];
  Areas2 = [stats2.Area];
  allCentroids1=[stats1(Areas1>5 & Areas1<100).Centroid];
  allCentroids2=[stats2(Areas2>5 & Areas2<100).Centroid];
  centroids1=[allCentroids1(1:2:end).' allCentroids1(2:2:end).'];
  centroids2=[allCentroids2(1:2:end).' allCentroids2(2:2:end).'];
  figure(5)
  set(5,'Name','Bead Identification Image 1','NumberTitle','off');
  imagesc(im1), axis ij, axis image, colormap gray, title(['Threshold: ' num2str(pct) ' %'])
  hold on
  plot(centroids1(:,1),centroids1(:,2),'ro');
  hold off
  figure(6)
  set(6,'Name','Bead Identification Image 2','NumberTitle','off');
  imagesc(im2), axis ij, axis image, colormap gray, title(['Threshold: ' num2str(pct) ' %'])
  hold on
  plot(centroids2(:,1),centroids2(:,2),'ro');
  hold off
  disp(['valid bead percentage image 1: ' num2str(length(centroids1)/length(Areas1)*100)])
  disp(['valid bead percentage image 2: ' num2str(length(centroids2)/length(Areas2)*100)])
  disp('***')
  pct_input=input(['Enter threshold (0-100%), current value ' num2str(pct) '%: ']);
  if pct_input>0 & pct_input<100;
    pct=pct_input;
  end
end
close 4 5 6

% compute centroids in image 2 based on beads in image 1 and the use cross-correlation to improve match
centroids2=cpcorr(centroids1, centroids1, reg_im2, im1);    % find matching beads using cross-correlation

% prepare vectors
x=centroids1(:,1);
y=centroids1(:,2);
dx=centroids2(:,1)-centroids1(:,1);
dy=centroids2(:,2)-centroids1(:,2);
```

```
% absDisp=sqrt(dx.^2+dy.^2);
scale=10;      % scale for vector display

figure(7)
set(7,'Name','Displacement Vectors','NumberTitle','off')
imagesc(im1), colormap gray
hold on
quiver(x,y,dx*scale,dy*scale,0,'r');
axis ij, axis image
hold off
title(['Displacement Vectors (Scale ' num2str(scale) ')'])

figure(8)
set(8,'Name','Bead Match Validation','NumberTitle','off')
pixels=impixel(reg_im2, centroids2(:,1), centroids2(:,2));
pixels=pixels(:,1);
imagesc(reg_im2), axis ij, axis image, colormap gray
hold on
plot(centroids2(pixels>thr2,1),centroids2(pixels>thr2,2),'g+');
plot(centroids2(pixels<=thr2,1),centroids2(pixels<=thr2,2),'r+');
title('matched beads')
hold off

corr_inp=input('Manual correction (y/[n])','s');
if strcmp(lower(corr_inp),'y')
   % find valid matched beads
   valid_centroids1=centroids1(pixels>thr2,:);
   valid_centroids2=centroids2(pixels>thr2,:);

   % manual correction:
   h=cpselect(reg_im2, im1, centroids2(pixels<thr2,:), centroids1(pixels<thr2,:));

   input('\n**********************\n** Manual Correction ** \n**********************\n When
done - save results, then hit <return>')
   dispose(h)

   corr_centroids1=base_points;
   corr_centroids2=cpcorr(input_points, base_points, reg_im2, im1);
   clear base_points input_points

   new_centroids1=[valid_centroids1; corr_centroids1];
   new_centroids2=[valid_centroids2; corr_centroids2];
   xc=new_centroids1(:,1);
   yc=new_centroids1(:,2);
   dxc=new_centroids2(:,1)-new_centroids1(:,1);
   dyc=new_centroids2(:,2)-new_centroids1(:,2);

   figure(9)
   set(9,'Name','Displacement Vectors','NumberTitle','off')
   imagesc(im1), colormap gray
   hold on
   quiver(xc,yc,dxc*scale,dyc*scale,0,'r');
   axis ij, axis image
   hold off
   title(['Displacement Vectors Corrected(Scale ' num2str(scale) ')'])
end
```

```
return
% strain map:
[xi,yi]=meshgrid([1:size(im1,2)],[1:size(im1,1)]);      %interpolated coordinates
x=new_centroids1(:,1);
y=new_centroids1(:,2);
absDisp=sqrt(dxc.^2+dyc.^2);
aDispInt=griddata(x,y,absDisp,xi,yi,'linear');%interpolated results
figure(10)
imagesc(aDispInt), axis image, colorbar
```

Cell size analysis program "CellSize"

```
% cell size measurement program
% reads images and allows manual measuremtn of cell length and witdh
% as well as automatic computation of cell area, major and minor axis
% length
% Jan Lammerding
% August 25-26, 2003
clear
Scale = 0.6437;    % um/pixel for CoolSNAP 10x
% Scale = 0.324;   % um/pixel for CoolSNAP 20x

[filename, pathname]=uigetfile('*.*','Select image');
if filename
   disp(['file: ' filename])
   file=fullfile(pathname,filename);
   imfile=filename(1:end-4);
   Im=imread(file);
   cd(pathname)
end

figure(1), imshow(Im), title('original image');

BWs = Im < 0.95 * mean2(Im);
%figure, imshow(BWs), title('binary threshold mask');

%BWs = edge(Im, 'sobel', (graythresh(Im) * .1));
%figure, imshow(BWs), title('binary gradient mask');

se90 = strel('line', 3, 90);
se0 = strel('line', 3, 0);

BWsdil = imdilate(BWs, [se90 se0]);
%figure, imshow(BWsdil), title('dilated gradient mask');

BWdfill = imfill(BWsdil, 'holes');
%figure, imshow(BWdfill); title('binary image with filled holes');

BWnobord = imclearborder(BWdfill, 4);
%figure, imshow(BWnobord), title('cleared border image');

seD = strel('diamond',1);
BWfinal = imerode(BWnobord,seD);
BWfinal = imerode(BWfinal,seD);
%BWfinal = imdilate(BWfinal,seD);
```

```
BWfinal = bwareaopen(BWfinal,1000,4);  % remove areas < 1000 pixel

figure(2), imshow(BWfinal), title('segmented image');

BWoutline = bwperim(BWfinal);
Segout = Im;
Segout(BWoutline) = 255;
figure(3), imshow(Segout), title('outlined original image');

% computation
L = bwlabel(BWfinal, 8);
n = max(max(L));
stats=regionprops(L, 'Area', 'Centroid', 'MajorAxisLength', 'MinorAxisLength', 'Orientation');
BWsize=[stats.Area];
temp = [stats.Centroid];
Cx = temp(1:2:end);
Cy = temp(2:2:end);
Orientation = [stats.Orientation];
Length = [stats.MajorAxisLength];
Width = [stats.MinorAxisLength];

% draw lines
%ang = Orientation/360*2*pi; % angles in radients
Lx1 = Cx - Length/2.*cos(Orientation/180*pi);
Lx2 = Cx + Length/2.*cos(Orientation/180*pi);
Ly1 = Cy + Length/2.*sin(Orientation/180*pi);
Ly2 = Cy - Length/2.*sin(Orientation/180*pi);

Lx3 = Cx + Width/2.*sin(Orientation/180*pi);
Lx4 = Cx - Width/2.*sin(Orientation/180*pi);
Ly3 = Cy + Width/2.*cos(Orientation/180*pi);
Ly4 = Cy - Width/2.*cos(Orientation/180*pi);

%select cells
N=0;
button = 0;
while button ~= 3
   title('select cell')
   [x, y, button] = ginput(1);
   x = median([1 x size(Im,2)]);
   y = median([1 y size(Im,1)]);
   cell = L(round(y),round(x));
   if cell > 0
      N=N+1;
      LL(N)=line([Lx1(cell) Lx2(cell)],[Ly1(cell) Ly2(cell)]);
      set(LL(N), 'Linewidth',2,'Color','r')
      LW(N)=line([Lx3(cell) Lx4(cell)],[Ly3(cell) Ly4(cell)]);
      set(LW(N), 'Linewidth',2,'Color','b')
      title('measure cell length')
      [x, y] = getline(3);
      x = x(1:2);
      y = y(1:2);
      LLm(N)=line(x, y); set(LLm(N), 'Linewidth',2,'Color','g');
      CellLengthMan(N)=sqrt(diff(x)^2 + diff(y)^2);
      CellLengthAut(N)=Length(cell);
```

```
    title('measure cell width')
    [x, y] = getline(3);
    x = x(1:2);
    y = y(1:2);
    LWm(N)=line(x, y); set(LWm(N), 'Linewidth',2,'Color','g');
    CellWidthMan(N)=sqrt(diff(x)^2 + diff(y)^2);
    CellWidthAut(N)=Width(cell);
    title('')
    CellArea(N)=BWsize(cell);
    txt(N) = text(Cx(cell)+100, Cy(cell),char(['Cell:   ' num2str(N)], ['Area:    ' num2str(CellArea(N))], ...
      ['Length: ' num2str(CellLengthMan(N))], ['Width:   ' num2str(CellWidthMan(N))]));
  end
end

%convert results into um
CellArea = CellArea * (Scale^2);
CellLengthMan = CellLengthMan * Scale;
CellLengthAut = CellLengthAut * Scale;
CellWidthMan = CellWidthMan * Scale;
CellWidthAut = CellWidthAut * Scale;

disp('*****************************************************************')
disp('**              Overall Results                    **')
disp('*****************************************************************')
disp(['File: ' imfile])
disp(sprintf('Cell\tSize (um2)\tLength (um)\tWidth (um)\tLong Axis (um)\tShort Axis (um)'))
for i=1:N

disp(sprintf('%3d\t%7.1f\t%7.2f\t%7.2f\t%7.2f\t%7.2f',i,CellArea(i),CellLengthMan(i),CellWidthMan(i),C
ellLengthAut(i),CellWidthAut(i)))
end
disp('-----------------------------------------------------------------')
disp(sprintf('mean\t%7.1f\t%7.2f\t%7.2f\t%7.2f\t%7.2f',mean(CellArea),mean(CellLengthMan),mean(Cell
WidthMan),mean(CellLengthAut),mean(CellWidthAut)));
disp(sprintf('stdev\t%7.1f\t%7.2f\t%7.2f\t%7.2f\t%7.2f',std(CellArea),std(CellLengthMan),std(CellWidthM
an),std(CellLengthAut),std(CellWidthAut)));
%disp(sprintf('mean\t%7.3f\t\t%7.2f\t\t\t%7.2f',mean(Nsize),mean(Nred),mean(Nblue)))
%disp(sprintf('stdev\t%7.3f\t\t%7.2f\t\t\t%7.2f',std(Nsize),std(Nred),std(Nblue)))
```

Analysis programs for displacement-time curves

Program "StepDataAnalysis"

```
% program to analyse and plot MPT results
% Jan Lammerding
% similar to dataanalysis, but for the modified force protocol, stepwise increasing force
% 01-31-03: modified line 138 to work in MATLAB R13

oldPath=cd;  %store old directory
scale=0.3; % microns/pixel, roper 30x
%scale=0.225; % roper 40x
fps=25; % frame rate
```

```
filename='default';
DataSet=0;
new=0;
time=[0:350]/fps;
while (filename | new)
  DataSet=DataSet+1;
  Xdata{DataSet}=[];
  XdispData{DataSet}=[];
  XResDispData{DataSet}=[];
  maxDispXData1{DataSet}=[];
  maxDispXData2{DataSet}=[];
  maxDispXData3{DataSet}=[];
  maxDispXData4{DataSet}=[];
  maxDispXData5{DataSet}=[];
  detachData{DataSet}=[];
  Legend{DataSet}=[];
  Results{DataSet}=[];
  n=0;
  nBeads{DataSet}=0;
  selection=0;
  filename='default';
  while filename
    [filename, pathname]=uigetfile('*.csv','Select MPT result file');
    if filename
      cd(pathname)
      file=fullfile(pathname,filename);
      [path, fname, ext, vers]=fileparts(file);
      FileData=csvread(file);
      [M,N]=size(FileData);
      if N>1 & M==351    %check if fileformat is valid
        n=n+1;
        if N>2
          str=[];
          beads=round(N/2);
          for i=1:beads
            str{i}=['bead ' num2str(i)];
          end
          if selection & selection<=beads    % use old selection as default
            def=selection;
          else
            def=1;
          end
          [selection,ok]=listdlg('PromptString','Select bead:', 'SelectionMode', 'single', 'Liststring', ...
            str,'InitialValue',def);
          if ok   % if bead has been selected
            beadX=FileData(:,selection*2-1)*scale;
            beadY=FileData(:,selection*2)*scale;
            beadStr=str{selection};
          else   % use first bead by default
            beadX=FileData(:,1)*scale;
            beadY=FileData(:,2)*scale;
            beadStr='bead 1';
          end
        else
          % only use first bead per file
          beadX=FileData(:,1)*scale;
```

```
  beadY=FileData(:,2)*scale;
  beadStr='bead 1';
end

% create label
k=findstr(fname,'_MPTresults');
if k>1
  label=[fname(1:k-1) ' - ' beadStr];
else
  label=[fname ' - ' beadStr];
end

% data analysis
InitialX=mean(beadX(20:45)); % use frames 20-45
max1X=mean(beadX(90:101));  % use last 10 frames of force application
max2X=mean(beadX(141:151)); % use last 10 frames of force application
max3X=mean(beadX(191:201)); % use last 10 frames of force application
max4X=mean(beadX(241:251)); % use last 10 frames of force application
max5X=mean(beadX(290:300)); % use last 10 frames of force application
FinalX=mean(beadX(341:351)); % use last 10 frames
Xdisp=beadX-InitialX;
if beadX(end)>0        % if bead is still attached at the end
  ResDisp=FinalX-InitialX;
else
  ResDisp=-1;
end

if beadX(101)>0
  maxDispX1=max1X-InitialX;
else maxDispX1=-1;
end
if beadX(151)>0
  maxDispX2=max2X-InitialX;
else maxDispX2=-1;
end
if beadX(201)>0
  maxDispX3=max3X-InitialX;
else maxDispX3=-1;
end
if beadX(251)>0
  maxDispX4=max4X-InitialX;
else maxDispX4=-1;
end
if beadX(301)>0
  maxDispX5=max5X-InitialX;
else maxDispX5=-1;
end
detach=min(find(beadX<0));
if isempty(detach)
  detach=-1;
end

% store information for series
Xdata{DataSet}=[Xdata{DataSet} beadX];
XdispData{DataSet}=[XdispData{DataSet} Xdisp];
XResDispData{DataSet}=[XResDispData{DataSet} ResDisp];
```

```
            maxDispXData1{DataSet}=[maxDispXData1{DataSet} maxDispX1];
            maxDispXData2{DataSet}=[maxDispXData2{DataSet} maxDispX2];
            maxDispXData3{DataSet}=[maxDispXData3{DataSet} maxDispX3];
            maxDispXData4{DataSet}=[maxDispXData4{DataSet} maxDispX4];
            maxDispXData5{DataSet}=[maxDispXData5{DataSet} maxDispX5];
            detachData{DataSet}=[detachData{DataSet} detach];
            Legend{DataSet}{n}=label;
            disp(['File: ' fname])
            disp([' max displacements: 0.3amp: ' num2str(maxDispX1) '  0.6amp: ' num2str(maxDispX2) ...
                '  0.9amp: ' num2str(maxDispX3) '  1.2amp: ' num2str(maxDispX4) ...
                '  1.5amp: ' num2str(maxDispX5) ' micron'])
            disp([' residual displ.:  ' num2str(ResDisp) ' micron'])
            disp([' detachment.:     ' num2str(detach) ' frame'])
            disp(' ')

            % store Results
            Results{DataSet}=[Results{DataSet}; maxDispX1 maxDispX2 maxDispX3 maxDispX4
maxDispX5 ResDisp detach];

            %plot results
            figure(DataSet)
            validData=XdispData{DataSet};
            validData(Xdata{DataSet}==-1*scale)=NaN;
            %plot(time,XdispData{DataSet})
            plot(time,validData);
            legend(Legend{DataSet},2)
            xlabel('time (s)')
            ylabel('x-displacement (\mum)')
            grid on

            %figure(DataSet+20)
            %plot(time,XdispData{DataSet},t+time(50),RegrData{DataSet},'r:')
            %legend(Legend{DataSet},2)
            %xlabel('time (s)')
            %ylabel('x-displacement (\mum)')
            %grid on
            %drawnow

        end
    end
  end

  if Xdata{DataSet}    % if we have results
    nBeads{DataSet}=n;
    new=1;
    % compute and display overall statistics (mean +- standard deviation):
    disp('*****overall statistics****')
    disp([' max displacement at 0.3amp (microns):  ' ...
        num2str(mean(maxDispXData1{DataSet}(maxDispXData1{DataSet}>0))) ' +-' ...
        num2str(std(maxDispXData1{DataSet}(maxDispXData1{DataSet}>0))) ])
    disp([' max displacement at 0.6amp (microns):  ' ...
        num2str(mean(maxDispXData2{DataSet}(maxDispXData2{DataSet}>0))) ' +-' ...
        num2str(std(maxDispXData2{DataSet}(maxDispXData2{DataSet}>0))) ])
    disp([' max displacement at 0.9amp (microns):  ' ...
        num2str(mean(maxDispXData3{DataSet}(maxDispXData3{DataSet}>0))) ' +-' ...
        num2str(std(maxDispXData3{DataSet}(maxDispXData3{DataSet}>0))) ])
```

```
    disp([' max displacement at 1.2amp (microns):  ' ...
        num2str(mean(maxDispXData4{DataSet}(maxDispXData4{DataSet}>0))) ' +-' ...
        num2str(std(maxDispXData4{DataSet}(maxDispXData4{DataSet}>0))) ])
    disp([' max displacement at 1.5amp (microns):  ' ...
        num2str(mean(maxDispXData5{DataSet}(maxDispXData5{DataSet}>0))) ' +-' ...
        num2str(std(maxDispXData5{DataSet}(maxDispXData5{DataSet}>0))) ])

    disp([' residual displ. (microns):          ' ...
        num2str(mean(XResDispData{DataSet}(XResDispData{DataSet}>0))) ' +-' ...
        num2str(std(XResDispData{DataSet}(XResDispData{DataSet}>0))) ])
    disp([' detachment (frame):                 ' ...
        num2str(mean(detachData{DataSet}(detachData{DataSet}>0))) ' +-' ...
        num2str(std(detachData{DataSet}(detachData{DataSet}>0))) ])
    disp([' # of beads detaching:               ' num2str(length(find(detachData{DataSet}>0)))])
    disp([' # of beads sticking:                ' num2str(length(find(detachData{DataSet}<0)))])
    disp('------------------------------------------------------------------------')
  else
    new=0;
  end
end

figure(100)
for i=1:DataSet-1
  validData=XdispData{i};
  validData(Xdata{i}==-1*scale)=NaN;
  subplot(DataSet-1,1,i)
  %plot(time,XdispData{i})
  plot(time,validData);
  legend(Legend{i},-1)
  xlabel('time (s)')
  ylabel('x-displacement (\mum)')
  grid on
end


% save results
[filename, pathname]=uiputfile('*.txt','Save results');
if filename
  %    [path, fname, ext, vers]=fileparts(filename);
  % use better extension check
  file=fullfile(pathname,[filename '.txt']);

  fid=fopen(file,'w');

  fprintf(fid,'\noverall statistics (mean +- standard deviation)');
  fprintf(fid,'\nbead \tmax displ. (0.3amp) \t \tmax displ. (0.6amp) \t \tmax displ. (0.9amp) \t \tmax displ.
(1.2amp) \t \tmax displ. (1.5amp) \t \tres. displ. \t \tdetachment (frame)');
  for i=1:DataSet-1
    fprintf(fid,'\n%s',Legend{i}{1}([1:end-10, end-6:end]));

fprintf(fid,'\t%7.3f\t%7.3f',mean(maxDispXData1{i}(maxDispXData1{i}>0)),std(maxDispXData1{i}(max
DispXData1{i}>0)));

fprintf(fid,'\t%7.3f\t%7.3f',mean(maxDispXData2{i}(maxDispXData2{i}>0)),std(maxDispXData2{i}(max
DispXData2{i}>0)));
```

```
fprintf(fid,'\t%7.3f\t%7.3f',mean(maxDispXData3{i}(maxDispXData3{i}>0)),std(maxDispXData3{i}(max
DispXData3{i}>0)));

fprintf(fid,'\t%7.3f\t%7.3f',mean(maxDispXData4{i}(maxDispXData4{i}>0)),std(maxDispXData4{i}(max
DispXData4{i}>0)));

fprintf(fid,'\t%7.3f\t%7.3f',mean(maxDispXData5{i}(maxDispXData5{i}>0)),std(maxDispXData5{i}(max
DispXData5{i}>0)));


fprintf(fid,'\t%7.3f\t%7.3f',mean(XResDispData{i}(XResDispData{i}>0)),std(XResDispData{i}(XResDis
pData{i}>0)));

fprintf(fid,'\t%7.3f\t%7.3f',mean(detachData{i}(detachData{i}>0)),std(detachData{i}(detachData{i}>0)));
  end

  fprintf(fid,'\n\n\ndetailed results');
  fprintf(fid,'\nbead \tmax displ. (0.3amp) \tmax displ. (0.6amp) \tmax displ. (0.9amp) \tmax displ.
(1.2amp) \tmax displ. (1.5amp) \tres. displ. \tdetachment (frame)');
  for i=1:DataSet-1
    for j=1:nBeads{i}
      fprintf(fid,'\n%s',Legend{i}{j});
      fprintf(fid,'\t%7.3f\t%7.3f\t%7.3f\t%7.3f\t%7.3f\t%7.3f\t%7.3f',Results{i}(j,:));
    end
  end

  fclose(fid);
end
%   fprint('-f1','-dtiff', fname[1:end-1]) save image
cd(oldPath);
return
```

Program "DataAnalysis" for force pulse data analysis

```
% program to analyse and plot MPT results
% Jan Lammerding
% 01/06/02
% 03/13/02: changed initial position: use frames 20-45
oldPath=cd;  %store old directory
scale=0.3; % microns/pixel (roper 30x)
%scale=0.225; % roper 40x
fps=25; % frame rate

filename='default';
DataSet=0;
new=0;
time=[0:200]/fps;
while (filename | new)
  DataSet=DataSet+1;
  Xdata{DataSet}=[];
  XdispData{DataSet}=[];
  XResDispData{DataSet}=[];
  maxDispXData{DataSet}=[];
  SlopeData{DataSet}=[];
```

```
RegrData{DataSet}=[];
Legend{DataSet}=[];
%files{DataSet}=[];
Results{DataSet}=[];
n=0;
nBeads{DataSet}=0;
selection=0;
filename='default';
while filename
  [filename, pathname]=uigetfile('*.csv','Select MPT result file');
  if filename
    cd(pathname)
    file=fullfile(pathname,filename);
    [path, fname, ext, vers]=fileparts(file);
    FileData=csvread(file);
    [M,N]=size(FileData);
    if N>1 & M==201    %check if fileformat is valid
      n=n+1;
      if N>2
        str=[];
        beads=round(N/2);
        for i=1:beads
          str{i}=['bead ' num2str(i)];
        end
        if selection & selection<=beads    % use old selection as default
          def=selection;
        else
          def=1;
        end
        [selection,ok]=listdlg('PromptString','Select
bead:','SelectionMode','single','Liststring',str,'InitialValue',def);
        if ok   % if bead has been selected
          beadX=FileData(:,selection*2-1)*scale;
          beadY=FileData(:,selection*2)*scale;
          beadStr=str{selection};
        else    % use first bead by default
          beadX=FileData(:,1)*scale;
          beadY=FileData(:,2)*scale;
          beadStr='bead 1';
        end
      else
        % only use first bead per file
        beadX=FileData(:,1)*scale;
        beadY=FileData(:,2)*scale;
        beadStr='bead 1';
      end

      % create label
      k=findstr(fname,'_MPTresults');
      if k>1
        label=[fname(1:k-1) ' - ' beadStr];
      else
        label=[fname ' - ' beadStr];
      end
    %   files{DataSet}=[files{DataSet}; label];
```

```
        % data analysis
        Prange=[76:125]; % data range to determine slope by fitting second order polynomial
        InitialX=mean(beadX(20:45)); % use frames 20-45
        FinalX=mean(beadX(191:201)); % use last 10 frames
        maxX=mean(beadX(115:125)); % use last 10 frames of force application
        Xdisp=beadX-InitialX;
        ResDisp=FinalX-InitialX;
        maxDispX=maxX-InitialX;
        Pcoeff=polyfit(time(Prange),Xdisp(Prange).',1);   % fit polynomial to last second of force
application

        % Regression for each bead
        % fit: dx=1/k0(1-(k1/(k0+k1)*exp(-t/(g1*(k0+k1)/(ko+k1)))))+t/g0, where coeff=(k0 k1 g0 g1)
%           fitErr=inline('norm(1/coeff(1)*(1-(coeff(2)/(coeff(1)+coeff(2)).*exp(-
t(:)/(coeff(4)*(coeff(1)+coeff(2))/(coeff(1)*coeff(2))))))+t(:)/coeff(3)-y)','coeff','t','y');
        fitErr=inline('norm(1/abs(coeff(1))*(1-(abs(coeff(2))/(abs(coeff(1))+abs(coeff(2))).*exp(-
t(:)/(abs(coeff(4))*(abs(coeff(1))+abs(coeff(2)))/(abs(coeff(1))*abs(coeff(2)))))))+t(:)/abs(coeff(3))-
y)','coeff','t','y');
        options=optimset('MaxIter',1e4,'MaxFunEvals',1e4);
        %fitCoeff=fminsearch(fitErr,[1 1 1 1],options,time(50:125)-time(50),Xdisp(50:125));
        fitCoeff=abs(fminsearch(fitErr,[1 1 1 1],options,time(50:125)-time(50),Xdisp(50:125)));
        t=time(50:125)-time(50);
        Regr=1/fitCoeff(1)*(1-(fitCoeff(2)/(fitCoeff(1)+fitCoeff(2)).*exp(-
t(:)/(fitCoeff(4)*(fitCoeff(1)+fitCoeff(2))/(fitCoeff(1)*fitCoeff(2))))))+t(:)/fitCoeff(3);

        % store information for series
        Xdata{DataSet}=[Xdata{DataSet} beadX];
        XdispData{DataSet}=[XdispData{DataSet} Xdisp];
        XResDispData{DataSet}=[XResDispData{DataSet} ResDisp];
        maxDispXData{DataSet}=[maxDispXData{DataSet} maxDispX];
        SlopeData{DataSet}=[SlopeData{DataSet} Pcoeff(1)];
        RegrData{DataSet}=[RegrData{DataSet} Regr];

        Legend{DataSet}{n}=label;
        disp(['File: ' fname])
        disp(['  max displacement: ' num2str(maxDispX) ' micron'])
        disp(['  residual displ.:  ' num2str(ResDisp) ' micron'])
        disp(['  creep response:   ' num2str(Pcoeff(1)) ' micron/sec'])
        disp(['  Bausch model coeffs (k0 k1 gamma0 gamma1):    ' num2str(fitCoeff) ])
        disp(' ')

        % store Results
        Results{DataSet}=[Results{DataSet}; maxDispX ResDisp Pcoeff(1) fitCoeff];

        % plot results
        %figure(DataSet)
        %plot(time,XdispData{DataSet})
        %legend(Legend{DataSet},2)
        %xlabel('time (s)')
        %ylabel('x-displacement (\mum)')
        %grid on

        figure(DataSet+20)
        plot(time,XdispData{DataSet},t+time(50),RegrData{DataSet},'r:')
        legend(Legend{DataSet},2)
        xlabel('time (s)')
```

```
            ylabel('x-displacement (\mum)')
            grid on
            drawnow

          end
        end
      end
      if Xdata{DataSet}    % if we have results
        nBeads{DataSet}=n;
        new=1;
        % compute and display overall statistics (mean +- standard deviation):
        disp('*****overall statistics****')
        disp([' max displacement (microns):   ' num2str(mean(maxDispXData{DataSet})) ' +-'
num2str(std(maxDispXData{DataSet})) ])
        disp([' residual displ. (microns):    ' num2str(mean(XResDispData{DataSet})) ' +-'
num2str(std(XResDispData{DataSet})) ])
        disp([' creep response (microns/sec): ' num2str(mean(SlopeData{DataSet})) ' +-'
num2str(std(SlopeData{DataSet})) ])
        disp('------------------------------------------------------------------------')
      else
        new=0;
      end
    end

    figure(100)
    for i=1:DataSet-1
      subplot(DataSet-1,1,i)
      plot(time,XdispData{i})
      legend(Legend{i},-1)
      xlabel('time (s)')
      ylabel('x-displacement (\mum)')
      grid on
    end


    % save results
    [filename, pathname]=uiputfile('*.txt','Save results');
    if filename
      %    [path, fname, ext, vers]=fileparts(filename);
      % use better extension check
      file=fullfile(pathname,[filename '.txt']);

      fid=fopen(file,'w');

      fprintf(fid,'\noverall statistics (mean +- standard deviation)');
      fprintf(fid,'\nbead \tmax displ. \t \tres. displ. \t \tslope');
      for i=1:DataSet-1
        fprintf(fid,'\n%s',Legend{i}{1}([1:end-10, end-6:end]));
        fprintf(fid,'\t%7.3f\t%7.3f',mean(maxDispXData{i}),std(maxDispXData{i}));
        fprintf(fid,'\t%7.3f\t%7.3f',mean(XResDispData{i}),std(XResDispData{i}));
        fprintf(fid,'\t%7.3f\t%7.3f',mean(SlopeData{i}),std(SlopeData{i}));
      end

      fprintf(fid,'\n\n\ndetailed results');
      fprintf(fid,'\nbead \tmax. displ. \tresid. displ. \tslope \tk0 \tk1 \tgamma0 \tgamma1');
      for i=1:DataSet-1
```

```
      for j=1:nBeads{i}
         fprintf(fid,'\n%s',Legend{i}{j});
         fprintf(fid,'\t%7.3f\t%7.3f\t%7.3f\t%7.3f\t%7.3f\t%7.3f\t%7.3f',Results{i}(j,:));
      end
   end

   fclose(fid);
end

%   fprint('-f1','-dtiff', fname[1:end-1]) save image
cd(oldPath);
return

function err2=Bauschfunct(C,r,urThetaData,Fo)
% fit the bead displacement in radial direction (normalized to cos(Theta)
% to the equation from Bausch-Membrane Model:
% ur(r) = Fo/(2*pi*mu) * cos(theta)*(...)
% Parameters to be fitted:
% C(1) = mu
% C(2) = kappa
% used for PSanalysis
sigma = 0.5;
mu = C(1);
k = C(2);
k1 = sqrt((1-sigma)/2)*k;
urTheta = Fo/(2*pi*mu) * (3/4*(1-sigma) * besselk(0,k1*r) - besselk(1,k*r)./(k*r) + sqrt((1-
sigma)/2)*besselk(1,k1*r)./(k*r));
% compute sum of squared error
err2=sum((urTheta-urThetaData).^2);
```

Program "SineMasterSelect" for sinusoidal forcing function

```
function SineMaster
% program to analyze Sine-wave data obtained from MPTD program
% based on function SinAnalysis from 8-16-02
% Jan Lammerding
% 08-24-02
% 08-27-02
% improved output format
% added save output option
% allow to open several files
% 8-29-02
% included option to include file in results or not or show more details
% added xls-friendly output of displacement data
% 8-30-02
% added residual displacement measurement
% 9-17-02: updated sine-force profile, i.e. 2sec relaxation at the end
% to do
% add option of drift correction or not
% use start menu for parameter settings or create GUI
% 6/12/03
% added FFT computed amplitude, updated output

oldPath=cd;  %store old directory
% make this interactive:
```

```
scale=0.3; % microns/pixel, roper 30x
%scale=0.225; % roper 40x
fps=60; % frame rate
ampl=1;  % force amplitude (amp)
displOn=0;  % option to display phase details

PhaseLagi=[];
PhaseLag=[];
Fdisplacement=[];
Fforce=[];
FFTampl=[];
MinDisp=[];
MaxDisp=[];
MeanDisp=[];
ResDisp=[];
DispAmp=[];
mAmp=[];
Xdisp=[];
Force=[];
Legend=[];
n=0;
filename='default';
while filename
  [filename, pathname]=uigetfile('*.csv','Select MPTD result file');
  if filename
    cd(pathname)
    file=fullfile(pathname,filename);
    [path, fname, ext, vers]=fileparts(file);
    FileData=csvread(file);
    [M,N]=size(FileData);
    if N>3
      warning('Only the first bead will be used !')
    end
    t=[0:M-1]./fps;
    x=FileData(:,1);      % raw x-position data
    %correct x-data for drift:
    pfit=polyfit(t(1:fps),x(1:fps)-x(1),1);
    drift=pfit(1)*t+pfit(2);      % drift of stage or bead
    x0=mean(x(round(fps/2):fps-1)); % initial bead position, evaluated last half a second before force
application
    xcd=x-drift;                % drift corrected x-position
    dxd=(xcd-xcd(1))*scale;       % relative displacement converted into microns for drift correction
    dx=(x-x0)*scale;              % relative displacement converted into microns, no drift correction
    %dx=dxd;      % use drift correction
    f=FileData(:,end);     % raw magnetic field data
    f=f/max(f)*2*ampl;  % scale magnetic data to force or amp

    figure(1)
    subplot(2,1,1)
    plot(t,x,'r',t,drift+x(1),'g:',t,xcd,'b'), legend('x-position','drift','corrected x-pos',2), ylabel('pixel')
    subplot(2,1,2)
    plot(t,dx,'r',t,dxd,'b'), legend('x-displ. (raw)','x-displ. (corr)',2), ylabel('um')
    figure(2)
    tf=[0:length(f)-1]/fps;
    plot(t,dx,'g',tf,f,'r'), legend('displacement','force',2), xlabel('time'),ylabel('force / displacemet')
    drawnow
```

```
    % PART II: DATA ANALYSIS:
    [phaseshift, fsignal, fforce, fftAmpl, Cmin, Cmax, Cmean, Amplitudes] = SinAnalysisFunc(dx, f,
scale, fps, displOn);
    %[phaseshift, fsignal, fforce, fftAmpl, Cmin, Cmax, Cmean, Amplitudes] = SinAnalysisFunc(dxd, f,
scale, fps, displOn);   % drift corrected

    % DISPLAY RESULTS
    disp('*****************************************************************')
    disp(sprintf('File:\t%s',fname))
    disp(sprintf('Displacement frequency:\t%7.3f\t\tForce frequency:\t%7.3f\t\t(FFT-
Analysis)',fsignal(1),fforce(1)))
    disp(sprintf('Displacement frequency:\t%7.3f\t\tForce frequency:\t%7.3f\t\t(X-corr-
Analysis)',fsignal(2),fforce(2)))
    disp(sprintf('Phase lag displacement:\t%7.3f\t\tPhase lag (intp.):\t%7.3f',phaseshift(1),phaseshift(2)))
    disp(sprintf('Amplitude at %4.2f Hz: \t%7.3f um (FFT with upsampled signal)',fsignal(3),fftAmpl))
    disp(sprintf('Residual displacement: \t%7.3f um',mean(dx(end-10:end))))
    disp(sprintf('Minima:    \t\t%s',sprintf('\t%7.3f',Cmin)))
    disp(sprintf('Maxima:    \t\t%s',sprintf('\t%7.3f',Cmax)))
    disp(sprintf('Means:     \t\t%s',sprintf('\t%7.3f',Cmean)))
    disp(sprintf('Amplitudes: \t\t%s',sprintf('\t%7.3f',Amplitudes)))

    button=questdlg('Accept current file ?','Sine-wave analysis','Accept','Show details','Reject','Accept');
    if strcmp(button,'Accept')
      n=n+1;
      PhaseLagi(n)=phaseshift(2);
      PhaseLag(n)=phaseshift(1);
      Fdisplacement(n)=fsignal(2);
      Fforce(n)=fforce(2);
      FFTampl(n)=fftAmpl;
      MinDisp{n}=Cmin;
      MaxDisp{n}=Cmax;
      MeanDisp{n}=Cmean;
      ResDisp(n)=mean(dx(end-10:end));      % residual displacement
      DispAmp{n}=Amplitudes;
      mAmp(n)=mean(Amplitudes);
      Xdisp(n,:)=dx(:).';
      Force(n,:)=f(:).';
      if findstr(fname,'_MPTresults')
        Legend{n}=fname(1:end-11);
      else
        Legend{n}=fname;
      end
      disp('Accepted')
    elseif strcmp(button,'Show details')
      [phaseshift, fsignal, fforce, fftAmpl, Cmin, Cmax, Cmean, Amplitudes] = SinAnalysisFunc(dx, f,
scale, fps, 1);
      disp('Detailed analysis - Press key to continue, then select file again to include it in results!')
      pause
      for i=10:16          % close windows
        if ishandle(i)
          close(i)
        end
      end
    else
      disp('Rejected')
```

```
    end
  end
end
% ANALYSIS OF ALL RESULTS
if n   % present total results
  figure(3)
  %subplot(2,1,1)
  plot(t,Xdisp), legend(Legend,-1), ylabel('x-displacement (um)'), xlabel('time (s)')
  figure(4)
  %subplot(2,1,2)
  plot(t,Force), legend(Legend,-1), ylabel('force'), xlabel('time (s)')

  disp('*****************************************************************')
  disp('**              Overall Results                    **')
  disp('*****************************************************************')
  % disp('Phaseshift:')
  disp(sprintf('frequency\tphase lag (interp.)\tphase lag\tFFT-ampl.\tmean ampl.\tresidual displ.\tfile'))
  for i=1:n

disp(sprintf('%7.3f\t\t\t\t%7.3f\t\t%7.3f\t\t%7.3f\t\t%7.3f\t\t%7.3f\t\t%s',Fforce(i),PhaseLagi(i),PhaseLag(i)
,FFTampl(i),mAmp(i),ResDisp(i),Legend{i}))
  end
  disp('-----------------------------------------------------------------')

disp(sprintf('%7.3f\t\t\t\t%7.3f\t\t%7.3f\t\t%7.3f\t\t%7.3f\t\t%7.3f\t\t%7.3f\t\t%s',mean(Fforce),mean(PhaseLagi),m
ean(PhaseLag),mean(FFTampl),mean(mAmp),mean(ResDisp),'Mean'))

disp(sprintf('%7.3f\t\t\t\t%7.3f\t\t%7.3f\t\t%7.3f\t\t%7.3f\t\t%7.3f\t\t%7.3f\t\t%s',std(Fforce),std(PhaseLagi),std(Pha
seLag),std(FFTampl),std(mAmp),std(ResDisp),'Stdv'))

  % save results
  [filename, pathname]=uiputfile('*.txt','Save results');
  if filename
    [path, fname, ext, vers]=fileparts(filename);
    file=fullfile(pathname,[fname '.txt']);
    fid=fopen(file,'w');
    fprintf(fid,'Sine-wave Analysis');
    fprintf(fid,'\nfile\tfrequency\tphase lag (interp.)\tphase lag\tFFT-ampl.\tmean ampl.\tresidual displ.');
    for i=1:n

fprintf(fid,'\n%s\t%7.3f\t%7.3f\t%7.3f\t%7.3f\t%7.3f\t%7.3f',Legend{i},Fforce(i),PhaseLagi(i),PhaseLag(i)
,FFTampl(i),mAmp(i),ResDisp(i));
    end

fprintf(fid,'\n\n%s\t%7.3f\t%7.3f\t%7.3f\t%7.3f\t%7.3f\t%7.3f','Mean',mean(Fforce),mean(PhaseLagi),mea
n(PhaseLag),mean(FFTampl),mean(mAmp),mean(ResDisp));

fprintf(fid,'\n%s\t%7.3f\t%7.3f\t%7.3f\t%7.3f\t%7.3f\t%7.3f','Stdv',std(Fforce),std(PhaseLagi),std(PhaseLa
g),std(FFTampl),std(mAmp),std(ResDisp));
    fprintf(fid,'\n\n%s\t%s\t%s\t%s\t%s','Mean',['=average(B3:B' num2str(n-1+3) ')'],
['=average(C3:C' num2str(n-1+3) ')'], ...
      ['=average(D3:D' num2str(n-1+3) ')'], ['=average(E3:E' num2str(n-1+3) ')'], ['=average(F3:F'
num2str(n-1+3) ')']);
    fprintf(fid,'\n%s\t%s\t%s\t%s\t%s','Stdev',['=stdev(B3:B' num2str(n-1+3) ')'], ['=stdev(C3:C'
num2str(n-1+3) ')'], ...
```

```
        ['=stdev(D3:D' num2str(n-1+3) ')'], ['=stdev(E3:E' num2str(n-1+3) ')'], ['=stdev(F3:F' num2str(n-
1+3) ')']);

      fprintf(fid,'\n\n\nAmplitudes (um)\nfile');
      fprintf(fid,'\tCycle %d',[1:length(DispAmp{1})]);
      for i=1:n
        fprintf(fid,'\n%s',Legend{i});
        fprintf(fid,'\t%7.3f',DispAmp{i});
      end

      fprintf(fid,'\n\n\nMean Displacement (um)\nfile');
      fprintf(fid,'\tCycle %d',[1:length(MeanDisp{1})]);
      for i=1:n
        fprintf(fid,'\n%s',Legend{i});
        fprintf(fid,'\t%7.3f',MeanDisp{i});
      end

      fprintf(fid,'\n\n\nMin Displacement (um)\nfile');
      fprintf(fid,'\tCycle %d',[1:length(MinDisp{1})]);
      for i=1:n
        fprintf(fid,'\n%s',Legend{i});
        fprintf(fid,'\t%7.3f',MinDisp{i});
      end

      fprintf(fid,'\n\n\nMax Displacement (um)\nfile');
      fprintf(fid,'\tCycle %d',[1:length(MaxDisp{1})]);
      for i=1:n
        fprintf(fid,'\n%s',Legend{i});
        fprintf(fid,'\t%7.3f',MaxDisp{i});
      end
      fclose(fid);
      disp(['file save as ' fname '.txt'])

      %   fprint('-f1','-dtiff', fname[1:end-1]) save image

    end

    % save displacement results in Excel friendly format
    [filename, pathname]=uiputfile('*.xls','Save curves in Excel-friendly format');
    if filename
      [path, fname, ext, vers]=fileparts(filename);
      file=fullfile(pathname,[fname '.xls']);
      fid=fopen(file,'w');
      fprintf(fid,'Displacement Data (um)');
      fprintf(fid,'\ntime\tforce');
      for i=1:n
        fprintf(fid,'\t%s',Legend{i});
      end
      for i=1:length(t)
        fprintf(fid,'\n%7.3f\t%7.3f',t(i),mean(Force(:,i)));
        fprintf(fid,'\t%7.3f',Xdisp(:,i));
      end
      fclose(fid);
      disp(['Displacement data save as ' fname '.xls'])
    end
end
```

```
cd(oldPath)

return


% SUBFUNCTIONS
function [phaseshift, fsignal, fcontrol, fftAmpl, Cmin, Cmax, Cmean, Amplitudes] = SinAnalysisFunc(x, f,
scale, fs, displ);
% Program to analyze sinwave data
% [phaseshift, fsignal, fcontrol, Cmin, Cmax, Cmean, amplitudes] = SinAnalysis(x, c, scale, fs, displ);
%  phaseshift: Phase shift (delay) of signal to control in seconds ([x-correlation-result; interpolated-result]
%  fsignal:   Signal frequency ([FFT-result; autocorrelation-result]
%  fcontrol:  Control frequency ([FFT-result; autocorrelation-result]
%  Cmin:      Minimum of each cycle
%  Cmax:      Maximum of each cycle
%  Cmean:     Mean of each cycle
%  amplitudes: Amplitudes of cycles, i.e. (Cmin-Cmax)/2
%  x:         signal (row vector), i.e. displacement in pixel
%  c:         control (row vector)
%  scale:     microscope scale: um/pixel
%  fs:        sampling frequence, i.e. frame rate (1 / sec)
%  displ:     display option (0 = off, 1 = on)


% test signals
%t=[0:600]/fs;
%delay=0.5*rand(1)
%sin1=0.5+0.5*sin(t*2*pi);
%sin2=0.5+0.5*sin((t-delay)*2*pi);
%x=sin2;
%f=sin1;
dx=x(:);       % convert to column vector
force=f(:);    % convert to column vector
time=[0:length(x)-1]/fs;   % time vector
Nfft = 1024;   % FFT order
ni=3;          % half width of interpolation window for phase delay

% process data
dxS=sgolayfilt(dx,3,15);       % smooth data

% cut out sinwave, use everything except for first and last two second
R = [fs:length(dx)-2*fs];    % valid signal indeces
dxR=dxS(R);              % valid (restricted) portion of smoothed displacement signal
timeR=time(R);           % valid portion of time signal
forceR=force(R);         % valid portion of force signal

% COMPUTE FREQUENCY
% 1) FFT approach:
%w = hamming(length(dxR));          % hamming window
w = hann(length(dxR));           % Hann Window
Y = fft((dxR-mean(dxR)).*w, Nfft);  % perform FFT on windows signal (mean corrected)
Pyy = Y.*conj(Y) / Nfft;          % compute power spectrum
Pyy = Pyy(1:(Nfft/2)+1);          % only use significant half
f = fs*(0:Nfft/2)/Nfft;          % frequency range
imin=round(0.2*Nfft/fs)+1;              % minimum index corresponding to minimum frequency of 0.2Hz
to exclude low frequency shifts
```

```
iPeak=find(Pyy(imin:end)==max(Pyy(imin:end)));        % find peak of power spectrum
fFFT=f(iPeak+imin-1);                  % find corresponding frequency

% repeat the same for the force
Yf = fft((forceR-mean(forceR)).*w, Nfft);  % perform FFT on windows signal (mean corrected)
Pyyf = Yf.*conj(Yf) / Nfft;          % compute power spectrum
Pyyf = Pyyf(1:(Nfft/2)+1);
iPeakf=find(Pyyf==max(Pyyf));         % find peak of power spectrum
fFFTf=f(iPeakf);                  % find corresponding frequency

% 2) autocorrelation approach
corr=xcorr(dxR-mean(dxR));            % auto-correlation of mean shifted displacement signal
midpt=ceil((length(corr)+1)/2);       % find center of autocorrelation function, i.e. delay=0
restrict=corr(midpt:end);             % only use positive delays
[pv,pind]=findpeak(restrict);            % find second maximum of autocorrelation function
fCorr = 1/(pind-1)*fs;                % compute delays
%repeat for force
corrf=xcorr(forceR-mean(forceR));
restrictf=corrf(midpt:end);
[pvf,pindf]=findpeak(restrictf);
fCorrf = 1/(pindf-1)*fs;

% find amplitude based on FFT and raw signal (upsampled by a factor Ni):
Ni=3;
x1=interp(dxR-mean(dxR),Ni);          % upsampled displacement, mean corrected
x2=interp(forceR-mean(forceR),Ni);     % upsampled force, mean corrected
Nfft1=2*floor(length(x1)/2);          % number of FFT points
Nfft2=2*floor(length(x2)/2);
x1=x1.*hann(length(x1));              % apply hanning window
x2=x1.*hann(length(x2));
y1=fft(x1,Nfft1); m1=abs(y1)/Nfft1*2;     % compute FFT and FFT magnitude for displacement
y2=fft(x1,Nfft2); m2=abs(y2)/Nfft2*2;     % compute FFT and FFT magnitude for force
f1 = (0:length(y1)-1)'*fs*Ni/Nfft1;       % frequency range
f2 = (0:length(y2)-1)'*fs*Ni/Nfft2;       % frequency range
%i1=find(m1==max(m1(1:Nfft1/2)));         % index of peak frequency of displacement signal
i2 = find(m2==max(m2(1:Nfft2/2)));        % index of peak frequency of forceR signal
i2 = max(i2,2*Ni+1);                  % make sure that index is in valid range
%Pfreq=sum(m1(i1-2*Ni:i1+2*Ni).^2);       % power (displacement signal) at peak frequency of
displacement signal
Pfreq=sum(m1(i2-2*Ni:i2+2*Ni).^2);        % power (displacement signal) at peak frequency of force
signal
%amplFFT1=sqrt(Pfreq/0.375)              % displacement signal amplitude at peak displacement frequency
amplFFT2=sqrt(Pfreq/0.375);             % displacement signal amplitude at peak force frequency

% compute phase shift
[crosscorr, L]=xcorr(dxR-mean(dxR),forceR-mean(forceR));
lcc=length(crosscorr);
%midpt=ceil((lcc+1)/2);
%restrictXC=crosscorr(midpt:end);
lags=L/fs;
iLpeak=find(crosscorr==max(crosscorr));
PhaseDelay=lags(iLpeak);

% use interpolation to improve resolution
lagsi=linspace(lags(iLpeak-ni),lags(iLpeak+ni));
Ccorri=interp1(lags(iLpeak-ni:iLpeak+ni),crosscorr(iLpeak-ni:iLpeak+ni),lagsi,'spline');
```

```
iLpeaki=find(Ccorri==max(Ccorri));
PhaseDelayi=lagsi(iLpeaki);


% compute amplitudes
lambda=round(1/fCorrf*fs);      % wave length in # of samples based on force autocorrelation
Nsections=floor(length(R)/lambda);  % number of full wave cycles
dxSection=zeros(Nsections,lambda);
Cmin=zeros(1,Nsections);
Cmax=zeros(1,Nsections);
Cmean=zeros(1,Nsections);
Amplitudes=zeros(1,Nsections);
for i=1:Nsections
   starti=1+(i-1)*lambda;
   endi=i*lambda;
   dxSection(i,:)=dxR(starti:endi).';
   % Cmin(i)=min(dxSection(i,max(PhaseDelay*fs,1):end));
   Cmin(i)=min(dxSection(i,:));
   Cmax(i)=max(dxSection(i,:));
   Cmean(i)=mean(dxSection(i,:));
   %   Cmed(i)=median(dxSection(i,:))
   Amplitudes(i)=(Cmax(i)-Cmin(i))/2;
end


% display results if display option is on
if displ
   % 1) data, smoothed data, and restricted data (during force application)
   figure(10)
   subplot(2,1,1)
   plot(time,dx,'g',time,dxS,'r',time,force,'b:')
   %legend('displacement','smoothed displacement','force',1)
   legend('displ','sm displ','force',-1)
   xlabel('time (s)'), ylabel('displacement (um) / force (nN)')
   subplot(2,1,2)
   plot(timeR,dxR,'r',timeR,forceR,'b:')
   legend('sm displ','force',-1)
   xlabel('time (s)'), ylabel('displacement (um) / force (nN)')


   % reference for 1 Hz signal
   % frame=[1:length(time)];
   % avidata=force;
   %test=max(avidata)/2+max(avidata)/2*sin((frame-61)/60*2*pi);
   %test(1:60)=0;
   %test(543:end)=0;
   %plot(frame,avidata,'g',frame,test,'r')


   % 2) Power Spectra of Data
   figure(11)
   subplot(2,1,1)
   plot(f,Pyy,'b',fFFT,Pyy(iPeak+imin-1),'ro')
   title('Frequency content of displacement')
   xlabel('frequency (Hz)')
   subplot(2,1,2)
```

```
  plot(f,Pyyf,'b',f(Pyyf==max(Pyyf)),max(Pyyf),'ro')
  title('Frequency content of force')
  xlabel('frequency (Hz)')

  % 3) auto-correlation of data
  figure(12)
  subplot(2,1,1)
  plot([0:length(restrict)-1]/fs, restrict)
  title('Auto-correlation of displacement')
  xlabel('delay (s)')
  subplot(2,1,2)
  plot([0:length(restrictf)-1]/fs, restrictf)
  title('Auto-correlation of force')
  xlabel('delay (s)')

  % Phase shift: cross-correlation
  figure(13)
  plot(lags,crosscorr,'g',lagsi,Ccorri,'r')
  title('Cross-correlation between force and displacement')
  xlabel('delay (s)')

  % Minima,Maxima and Amplitudes of each cycle
  figure(14)
  for i=1:Nsections
     subplot(Nsections,1,i)
     plot(i+[0:length(dxSection(i,:))-1]/60,dxSection(i,:),'g', ...
        i+(find(dxSection(i,:)==Cmin(i))-1)/fs,Cmin(i),'b+', ...
        i+(find(dxSection(i,:)==Cmax(i))-1)/fs,Cmax(i),'r+')
     legend(['cycle ' num2str(i)],-1), ylabel('displ (um)')
  end
  xlabel('time (s)')

  figure(15)
  n=[1:Nsections];
  subplot(2,1,1)
  plot(n,Cmin,'g',n,Cmin,'b+',n,Cmax,'g',n,Cmax,'r+',n,Cmean,'g:',n,Cmean,'c+')
  xlabel('cycle'), ylabel('min., max., and mean displacement')
  %  legend('min','max','mean')
  subplot(2,1,2)
  plot(n,Amplitudes,'g',n,Amplitudes,'b+'), xlabel('cycle'), ylabel('amplitude')

  figure(16)
  subplot(2,1,1)
  plot(f1(1:Nfft1/2+1),m1(1:Nfft1/2+1));    % frequency range
  title('Power spectrum of upsampled dx signal')
  subplot(2,1,2)
  plot(f1(1:Nfft2/2+1),m2(1:Nfft2/2+1));    % frequency range
  title('Power spectrum of upsampled force signal')

end
% return results
phaseshift = [PhaseDelay; PhaseDelayi];
fsignal = [ fFFT; fCorr; f2(i2)];
fftAmpl = amplFFT2;
fcontrol = [ fFFTf; fCorrf];
```

```
return

function [peakval,peakindex] = findpeak(x)
%PEAK Detects value and index of peak in autocorrelation function.
%       [peakval peakindex] = peak(x) locates the value and index of the
%       largest peak in the vector x other than Rx[0].  x must be an
%       autocorrelation function with maximum value Rx[0] its first element.
%  3/19/93   J. Greenberg
%  modified by J. Lammerding on 8-16-02 to allow non-zero crossing detection
if ( (size(x,1) > 1) & (size(x,2) > 1) )
  error ('Signal must be a vector')
end
x = x(:);
%if ( x(1,1) ~= max(x) )
% error('Input must be an autocorrelation function with Rx[0] as first element')
%end
nx = length(x);
flag = 0;
i = 2;
while (flag == 0)
  %  if ( x(i,1) < 0 )
  if x(i,1) >= x(i-1,1)
    flag = 1;
  else
    i = i+1;
  end
end
[peakval peakindex] = max(x(i:nx));
peakindex = peakindex+i-1;
return
```

Program "Calibration" for magnetic trap calibration analysis

```
% Program for force calibration of the single pole magnetic trap
% reads bead coordinates in csv-file format,
% smoothes curves,
% takes derivative and computes force vs distance relationship
% global regression coeff's contained in variable 'c2'
% indiv. regression coeff's contained in variable 'coeff'
% Author: Jan Lammerding
% Date: 15/05/01
%
% modified 1-26-03: manually enter trap location, compute drift force, simplify

% to do:
% option to save results

clear
% scale=0.3;                % scale [micron/pixel], Roper ES310T 30x
scale=0.4483                % scale [micron/pixel], Roper ES310T 20x
fps=25              % frame rate [fps per second]
% values for dimethylpolysiloxane
```

```
rho=0.974;                                          % density [g/ml]
kv=12500;                                           % kinematic viscosity
[centistokes]=[mm^2/s]
D=4.5;                                              % bead diameter [microns]
c=3*pi*D*rho*kv*1e-6;          %constant to convert velocity to force, according to stokes law: F=c*v
[nN]

tol_end=10;     % ignore beads within tol_end from magnetic trap
tol_start=2;   % ignore bead data that movement within tol_start from starting point (default: 2)
w=7;          % window width for smoothing
K=1;          % polynomial order for smoothing
  colors={'b','g','r','c','m','y','k','b:','g:','r:'};

% open dialog box to load file
[filename pathname]=uigetfile('*.csv','Open File with Coordinates');
if filename
  % read data from file
  data=csvread(fullfile(pathname,filename));
  % determine number of beads (each bead takes up two columns)
  [rows columns]=size(data);
  n=columns/2;
  beadlegend=[];
  % enter magnetic trap position and automatically detect end points from data
  for i=1:n
    % create beadlegend
    beadlegend=char([beadlegend; 'bead ', num2str(i)]);
    % extract bead coordinates from data
    beadx{i}=data(:,i*2-1);
    beady{i}=data(:,i*2);
    MTx(i)=input('Magnetic trap x-coordinate ...');
    figure(1)
    plot(beadx{i},beady{i},'+')
    d=(MTx(i)-beadx{i}); %compute distance of bead data point from mouse position
    %endN(i)=find(d==min(d(d>=tol_end/scale)));   % find the data point closest to mouse (and at least
'end_tol' away from trap)
    endN(i)=find(d==min(d(d>0)));   % find the data point closest to mouse
    plot(beadx{i},beady{i},'+',beadx{i}(endN(i)),beady{i}(endN(i)),'ro')
    title(['Bead ' num2str(i) ': Autmatic end point detection'])
  end
  pause(1)
  close(1)

  % determine length of valid data and create time axis
  maxN=max(endN);                                   % max frame# for slowest bead to reach final
position
  time=([1:maxN]./fps).'; %time vector [s]

  % compute drift force
  for i=1:n
    pfit=polyfit(time(1:fps),beadx{i}(1:fps)-beadx{i}(1),1);
    driftV(i)=pfit(1)*scale;      % drift of bead (velocity in nm/sec)
  end
  driftF=driftV*c;        % drift force (nN)

  % data processing
  for i=1:n
```

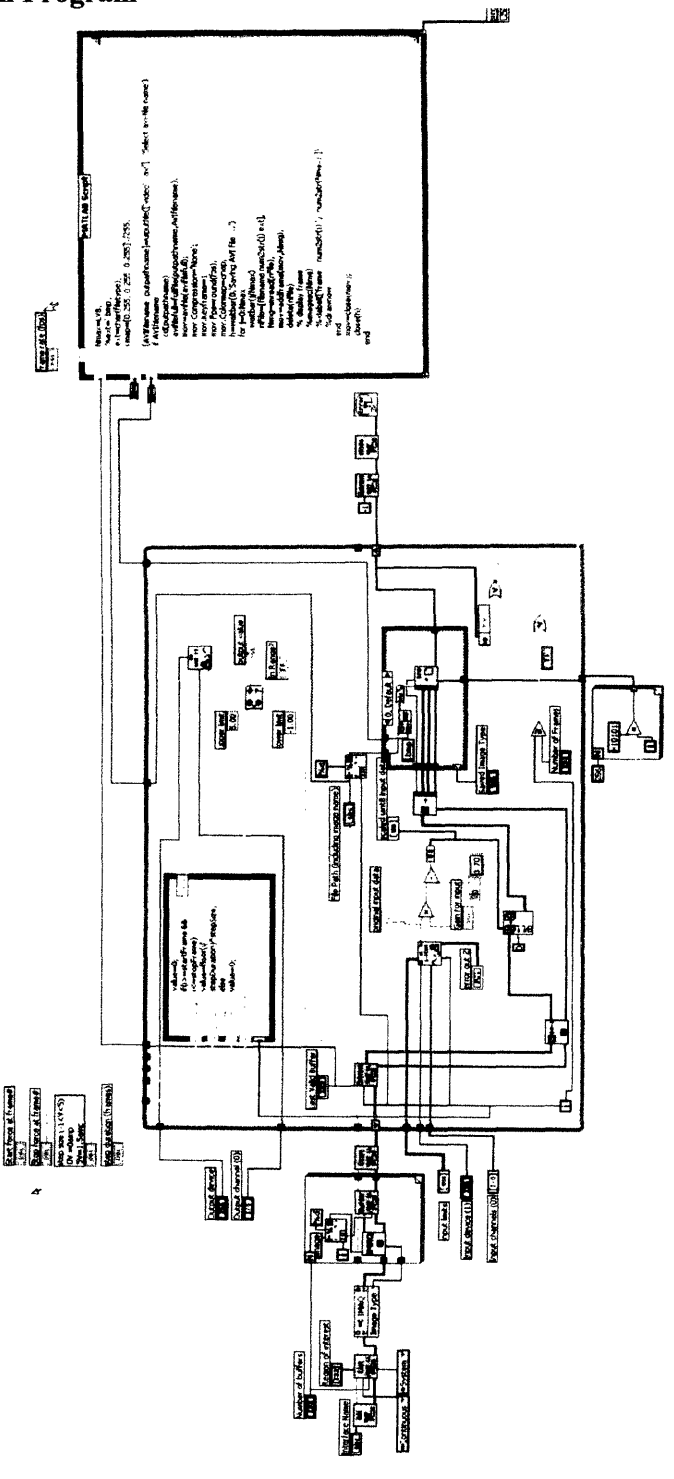
```
  % smooth positions using Savitzky-Golay (polynomial) FIR smoothing
  bx{i}=sgolayfilt(beadx{i}(1:endN(i)),K,w);
  % compute distance from end point, convert into microns
  dist=(MTx(i)-bx{i})*scale;
  %distance{i}=(MTx(i)-bx{i})*scale;
  start_d=mean(dist(1:10)); %compute starting point for each bead
  distance{i}=dist;
  %COMPUTE VELOCITY
  velocity{i}=-gradient(distance{i},1/fps);
  %COMPUTE FORCE
  force{i}=c*velocity{i};
  % apply tolerance criteria:
  valid_distance{i}=distance{i}(distance{i}>tol_end & distance{i}<start_d-tol_start);
  valid_force{i}=force{i}(distance{i}>tol_end & distance{i}<start_d-tol_start);
end
figure(1)
clf
hold on
for i=1:n
  plot(time(1:length(distance{i})),distance{i},colors{i})
end
hold off
xlabel('time [sec]')
ylabel('distance [micron]')
legend(beadlegend,0)

figure(2);
clf
hold on
for i=1:n
  plot(distance{i},force{i},colors{i})
end
hold off
title('force profile')
xlabel('distance [microns]')
ylabel('force [nN]')
legend(beadlegend,-1)
grid on

figure(3);
clf
hold on
for i=1:n
  plot(valid_distance{i},valid_force{i},colors{i})
end
hold off
title('valid force profile')
xlabel('distance [microns]')
ylabel('force [nN]')
legend(beadlegend,-1)
grid on

% compute maximum valid distance
maxDist=0;
for i=1:n
  if max(valid_distance{i})>maxDist
```

```
        maxDist=max(valid_distance{i});
    end
end

% Allow user to ignore one outlier for global regression
outlier=inputdlg('Enter outlier you wish to ignore','Calibration Program',1,{'0'});
if length(outlier)
    outlier=str2num(outlier{1});
    outlier=median([0 outlier n]);  % make sure that outlier is between 0 and bead#
else
    outlier=0;
end

acc_dist=[];   % initialize array containinig all bead data
acc_force=[];
figure(4)
for i=1:n;                  % for each bead
    % store all values for global regression
    if i~=outlier           % exclude outlier
        acc_dist=[acc_dist; valid_distance{i}];
        acc_force=[acc_force; valid_force{i}];
    end
    % Regression for each bead
    ForceErr=inline('norm(lambda(1)./(d+lambda(2)).^lambda(3)-y)','lambda','d','y');    %define error
function
    options=optimset('MaxIter',1e4,'MaxFunEvals',1e4);
    coeff(i,:)=fminsearch(ForceErr,[10 0 1],options,valid_distance{i},valid_force{i});
    forceRegr=coeff(i,1)./((valid_distance{i}+coeff(i,2)).^coeff(i,3)); % regression data

    subplot(n,1,i)
    plot(valid_distance{i},valid_force{i},'r',valid_distance{i},forceRegr,'b')
    legend(char(['Bead ' num2str(i)]),'regression',-1);
    ylabel('force [nN]')
    axis([0 maxDist 0 max(forceRegr)])
end
xlabel('distance [\mum]')

% Global regression for all beads:
c2=fminsearch(ForceErr,[10 0 1],options,acc_dist,acc_force);
dis=linspace(0, maxDist, 100);
GforceRegr=c2(1)./((dis+c2(2)).^c2(3));

figure(5)
plot(acc_dist,acc_force,'y+',dis,GforceRegr,'b')      % plots data points considered for regression
title('Global regression for force vs distance')
xlabel('distance [\mum]')
ylabel('force [nN]')
axis([tol_end maxDist 0 max(acc_force)])
grid on
legend('Data Points used for Regression','Regression')

% new display
disp('Curve fitting')
disp('fitted the following equation to the data')
disp('force=a/(distance+b)^n')
disp('coefficients for individual curve fit (coeff):')
```

```
    disp(sprintf('a\t\t\tb\t\t\tn'))
    for i=1:n
       disp(sprintf('%9.4f\t%9.4f\t%9.4f',coeff(i,:)))
    end
    disp(sprintf('\n\nCoefficients for global curve fit (c2):'))
    disp(sprintf('a\t\t\tb\t\t\tn'))
    disp(sprintf('%9.4f\t%9.4f\t%9.4f\n',c2))
    disp(sprintf('\ndrift force in [nN] (driftF)'))
    disp(sprintf('%7.3f\n',driftF))
end
```

Appendix G. LabView programs

LabVIEW Step-function Program



LabVIEW Sine-wave program