Using Virtual Articulations to Operate High-DoF Manipulation and Inspection Motions

The MIT Faculty has made this article openly available. Please share how this access benefits you. Your story matters.

<table>
<thead>
<tr>
<th>Citation</th>
<th>Vona, Marsette et al. &quot;Using Virtual Articulations to Operate High-DoF Inspection and Manipulation Motions&quot; International Conference on Field and Service Robotics, 7th, 2009.</th>
</tr>
</thead>
<tbody>
<tr>
<td>As Published</td>
<td><a href="http://www.rec.ri.cmu.edu/fsr09/papers/FSR2009_0033_a634a196d0287b318c16c563dee021fe.pdf">http://www.rec.ri.cmu.edu/fsr09/papers/FSR2009_0033_a634a196d0287b318c16c563dee021fe.pdf</a></td>
</tr>
<tr>
<td>Publisher</td>
<td>International Conference on Field and Service Robotics</td>
</tr>
<tr>
<td>Version</td>
<td>Author’s final manuscript</td>
</tr>
<tr>
<td>Citable link</td>
<td><a href="http://hdl.handle.net/1721.1/63171">http://hdl.handle.net/1721.1/63171</a></td>
</tr>
<tr>
<td>Terms of Use</td>
<td>Creative Commons Attribution-Noncommercial-Share Alike 3.0</td>
</tr>
<tr>
<td>Detailed Terms</td>
<td><a href="http://creativecommons.org/licenses/by-nc-sa/3.0/">http://creativecommons.org/licenses/by-nc-sa/3.0/</a></td>
</tr>
</tbody>
</table>
Using Virtual Articulations to Operate High-DoF Inspection and Manipulation Motions

Marsette Vona, David Mittman, Jeffrey S. Norris, and Daniela Rus

Abstract We have developed a new operator interface system for high-DoF articulated robots based on the idea of allowing the operator to extend the robot’s actual kinematics with virtual articulations. These virtual links and joints can model both primary task DoF and constraints on whole-robot coordinated motion. Unlike other methods, our approach can be applied to robots and tasks of arbitrary kinematic topology, and allows specifying motion with a scalable level of detail. We present hardware results where NASA/JPL’s All-Terrain Hex-Legged Extra-Terrestrial Explorer (ATHLETE) executes previously challenging inspection and manipulation motions involving coordinated motion of all 36 of the robot’s joints.

1 Introduction

Due to their application flexibility, robots with large numbers of joints are increasingly common: humanoids with 20 or more DoF are now available off-the-shelf, many-link serpentine robots have been demonstrated with a wide range of locomotion modalities, and assemblies of modular and self-reconfiguring hardware have been constructed with many 10s of concurrently active joints. This flexibility is especially attractive for interplanetary and Lunar exploration contexts, where the extreme costs of transportation from Earth are balanced by maximizing versatility, reusability, and redundancy in the delivered surface system hardware. Such considerations have been a prime motivation for NASA/JPL’s development of the 36-DoF All-Terrain Hex-Legged Extra-Terrestrial Explorer (ATHLETE) [13], with which astronauts will collaborate in our planned return to explore the Moon (figure 1).
Mission cost also dictates that we need operator interface systems that can rapidly and efficiently expose the maximum hardware capability to the humans that direct these robots, whether they are on-site astronauts or ground-based operators. This is a challenging problem in the high-DoF case: there are usually many ways the robot could move to achieve the task, and some may be better than others due to secondary goals. Sometimes a human operator can quickly visualize the desired motion, but till now the expression of this motion to the operations system has often been a tedious bottleneck. In this paper we present the design, implementation, and experimental results for a new operations system for high-DoF robots which employs virtual articulations to address this issue.

In our system, which we call the mixed real/virtual operator interface, the operator is presented with a graphical model of the robot and a palette of available joint types (figure 2, left). To constrain motion for a particular task, the operator instantiates virtual joints from this palette and interconnects them to the links of the actual robot and/or to new virtual links, constructing arbitrary virtual extensions to the actual robot kinematics. Virtual joints can be erected to parametrize specific task DoF; for example the long prismatic virtual joint in figure 4 parametrizes the length of a trenching motion. By closing kinematic chains, virtual articulations can also constrain whole-robot motion, thus narrowing the space of possible motions for a redundant task to those that satisfy the operator’s intentions. The virtual Cartesian-3 joint in figure 4, which allows three axes of translation but no rotation, constrains ATHLETE’s deck to remain flat, even while moving to extend reach for the primary trenching task. Virtual links can serve as interconnection points for more complex constructions of virtual joints—the chain of two prismatic and two revolute virtual joints in figure 4 is interspersed with three virtual links—and can also model task-related coordinate frames or world objects (figure 3).

Once virtual articulations are constructed for a task, the operator can move any joint or link (e.g. with the mouse), and the system interactively responds in real-time with a compatible motion for all joints which best satisfies all constraints. For example, in the trenching task, the operator can effectively command “trench from
-0.9m to +0.4m” by operating the corresponding virtual prismatic joint, or they may simply drag the constrained end effector with the mouse. We validate these motions in simulation and then execute them on the hardware.

Our system is generally applicable to kinematic operations in articulated robots of any topology, handling both open- and closed-chain constructions as well as both over- and under-constraint. In this paper we focus on our recent results operating new ATHLETE motions at JPL, but in other work we have also begun to demonstrate the usefulness of our approach in specifying motions for modular reconfigurable robots which can be assembled in arbitrary topologies. We expect that applications to other high-DoF kinematic motions, including in humanoids and in serpentine robots, will also be both useful and direct.

We describe related work next. Then we explain the architecture of our mixed real/virtual operator interface and detail the handling of under- and over-constrained cases, a key aspect of our system. Next we show several inspection and manipulation tasks on the ATHLETE hardware that would have been challenging with prior operator interfaces, including an experiment where we combine our virtual articulation interface with a direct-manipulation input device that mimics one ATHLETE limb. We developed this device, the Tele-Robotic ATHLETE Controller for Kinematics (TRACK), in prior work [6]. We conclude by summarizing known limitations of our approach and possible future directions.

2 Related Work

We see our new method of operating high-DoF robots using virtual articulations as filling a gap between existing low-level methods, including forward and inverse kinematic control, and existing high-level methods such as goal-based motion planning and programing-by-demonstration.

Bare kinematic control without higher-level goals or constraints is potentially tedious in the high-DoF case given the high dimension of the joint space. Task priority and task space augmentation approaches [7] can support high-DoF motion using holonomic constraints, but do not themselves offer any particular way to specify those constraints. Our virtual articulation approach addresses this with a concrete framework in which holonomic constraints can be constructed by an operator.

Goal-based motion planning, e.g. the classic “piano moving” problem of achieving a target configuration among obstacles, is typically not directly applicable in cases where the operator would also like to specify more detailed or continuous aspects of the motion. If we want such scalable motion specification, to constrain motion “on the way” to a primary goal configuration, we need something more. Virtual articulations are one language that does permit such scaling: the operator can constrain motion as much or as little as desired.

Programming-by-demonstration allows more specific motion specification, but is hard to apply when the robot topology diverges from preexisting systems and biology. Thus it has been used with some success for humanoids, or when mimicking
hardware is available, as in our prior work with the TRACK direct-manipulation hardware interface. But, short of building a full 36-DoF scale model, how to apply the technique to the whole ATHLETE mechanism, or in general, for arbitrary topology robots? Virtual articulations are not tied to any particular topology. Further, in section 4 we show that an integration of TRACK with our virtual articulations system can have some of the advantages of both.

Though we don’t find any prior authors using virtual articulations to build a general-purpose operations interface as we have done, there have been some related ideas. Virtual reality operator interfaces (e.g. [3]) have been explored where a model of the robot and its surroundings is provided to the operator for pose manipulation; we go beyond this by allowing the operator to virtually change and augment the kinematic structure. Our approach was motivated in part by past work with geometric constraints in graphics and animation [8, 12]; we show that a homogeneous model of only links and joints is sufficient in some practical applications.

Finally, we note that CAD constraint solvers [5] and physics simulators [10] have similar capabilities to our system. CAD solvers usually don’t permit over-constraint, and typically use heuristics to infer constraints in under-constrained cases, which may or may not be what the operator intended; our system usefully handles both under- and over-constraint without heuristics. Physics simulators can also be problematic in over-constrained cases, and the need to specify physics parameters such as mass and friction properties could make the process of building virtual articulations much more tedious. Our current approach is purely kinematic, so constructing virtual articulations only requires posing them in space and connecting them. Pratt et al explored a dynamic correlate to our virtual articulation approach which they called virtual model control [9] for some applications in legged locomotion.

3 The Mixed Real/Virtual Operator Interface

The key advance that differentiates our system from prior approaches is that we permit the operator to interactively construct virtual links and joints both to constrain and parametrize the primary task and also to constrain coordinated whole-robot motion. In this section we give an overview of the architecture of our system and explain how we address handling of under- and over-constrained cases, which are both common and important. Due to space constraints we omit our approaches to a number of other issues which do need to be considered in a full implementation, including: joint pose representation, handling of joint motion limits, efficient and accurate Jacobian computation, joint inversions and re-grounding, model complexity management, automatic decomposition of the model into independently solvable pieces, adaptive step size and damping, and graphics/UI features.

Figure 2 shows our system’s architecture. There are three categories of inputs: (1) robot models are loaded into the system from standard file formats such as VRML; (2) the operator may add and remove virtual articulations on-line with a variety of topological mutators; and (3) the operator may move joints and links, either virtual
or actual, by a variety of means including mouse and keyboard actions, waypoint sequencing and interpolation, or by using special-purpose input hardware.

These inputs determine the evolution of both topological and numeric models that include the actual robot kinematics plus any virtual articulations. The topological model is a kinematic graph where the edges correspond to joints and the vertices to links. The numeric model consists of the current pose of each joint as well as the joint constraints, motion limits, and several kinds of goals, as described below.

The main action of the system is to compute feasible whole-robot motions in response to the operator’s requests to move joints and links. Our approach is iterative local linear optimization based on the damped least squares Jacobian pseudoinverse and nullspace projection. This well-known approach has long been used in robotics; we apply a multi-priority formulation recently presented for a graphics application in [1]. The per-iteration time complexity of this formulation, typically dominated by a nullspace projection step, is quadratic in the number of joints. Nevertheless our implementation achieves real-time (several 10s of ms) response to operator motion gestures for the 50+ joints comprising the ATHLETE model plus various added virtual articulations. As with any local optimization approach, local optima must be avoided by higher-level means—the system is more of a controller than a planner. In our work thus far these global planning tasks are handled by the operator.

As feasible motions are computed they drive the two main outputs of the system: an interactive 3D graphical display of the robot plus any virtual articulations, and trajectories of the robot’s joints that can be sent to the hardware for execution.

In general the space of feasible motions may be continuous (under-constrained case), discrete (well-constrained), or empty (over-constrained). Since the well-
constrained case requires an exact balance of freedom and constraint, the under- and over-constrained cases are more common, and we give them special attention.

**Handling Under-Constraint** When the operator adds a virtual joint closing a kinematic chain the dimension of the feasible configuration space can be reduced. This is the first of three ways that we address under-constrained (aka redundant) problems: the operator may intentionally construct virtual articulations to express specific motion constraints and thus reduce redundancy.

The second way we handle redundancy is by exposing two levels of joint pose goals to the operator: targets and postures. A target in our system is the pose to which the operator has specifically manipulated (e.g. by mouse or keyboard interaction, or by waypoint interpolation) a joint or link. A posture models a default pose; for ATHLETE operations we typically set joint postures according to the “standard driving pose” (figure 1). The system solves first for motions which best attain all targets, and within the set of motions which do, the system second tries to attain postures. Target and posture are both optional for each DoF of each joint.

Goal attainment is prioritized in our system by structuring the solvers according the formulation presented in [1], which we call *prioritized damped least squares* (PDLS). In this formulation there are an arbitrary number of priority levels, each containing an arbitrary set of constraints. The constraints at the highest priority level are solved first, and the solution for each subsequent level is projected onto the nullspace of the levels above it.

The least squares aspect of PDLS provides the third and ultimate means of handling under-constraint. The least squares solution to an under-constrained problem will select a shortest step in joint space at each iteration, resulting in incrementally minimal motion: at a fine scale, the system will produce direct straight-line moves from one configuration to the next. In the under-constrained case a roundabout trajectory might also satisfy the constraints and maximize goal attainment, but would doubtless be surprising to the operator.

**Priority Levels and Handling Over-Constraint** The least-squares nature of PDLS also means that within a priority level, over-constraint will result in a solution which minimizes the squared error across all constraints in the level. This is useful and can produce intuitive behavior from the operator’s perspective. Another important feature of PDLS in over-constrained cases is the prioritization: satisfaction of constraints at a lower priority level will not compromise satisfaction at higher levels, even when the constraints conflict.

There are four priority levels in our system:

1. Joint invariants are solved at the highest priority level. For example, a spherical joint permits no translation, so when closing a kinematic chain it induces three invariant goals expressing that its three translation components must be zero.

---

1 To model pose goals on a link \( l \) we transparently introduce a virtual general (unconstrained 6-DoF) joint \( j \) connecting \( l \) to the world frame, and set the goals on \( j \).

2 And to complete the terminology, damping refers to the well-known technique of numeric stabilization at near-singular configurations by introducing a damping factor.
2. Lock goals model joints that have been “frozen” by the operator: each DoF of such a joint must remain as it was when the joint was first locked.

3. Target goals model intended joint and link poses as described above.

4. Posture goals model default poses, also described above.

It would also be possible to insert other (differentiable) optimality criteria, such as manipulability maximization or joint limit avoidance, as new priority levels.

To see how priority levels help in cases of over-constraint, consider the spherical object inspection task in figure 3. In this case we use TRACK to pose the limb holding the inspection camera. But there is also a virtual spherical joint constraining the camera, and TRACK has no haptic feedback. So, while the operator will generally try to pose it near to a feasible configuration, invariably this will diverge from the strict spherical constraint surface, over-constraining the limb. The spherical joint constraint is modeled at the invariant level, and TRACK’s pose is modeled at the target level, so the system will automatically sacrifice the latter for the former. The overall effect is as if the virtual spherical joint was physically present and rigidly constraining the motion, and as if there were an elastic connection between TRACK and the motion of the actual limb.

4 Operating ATHLETE with Virtual Articulations

The object inspection task is one of four hardware experiments we present. All show the ability of our mixed real/virtual interface system to help design specific motions which are rapid for human operators to conceptualize but difficult to express in prior operations interfaces, including several other software systems under development within NASA [4, 11] as well as our own TRACK device used alone [6].

![Fig. 3 ATHLETE inspecting an object using both the mixed real/virtual interface and TRACK, a special-purpose input device that mimics one limb.](image-url)
For the object inspection task, the operator designs a motion where a limb-mounted camera inspects a roughly spherical object while maintaining a constant distance. The operator directly models this constraint using a virtual spherical joint connecting the object (itself represented as a virtual link) and the camera. A secondary goal is to extend the space of reachable viewpoints by using the five other limbs to lean the hexagonal deck, but because the deck often carries a payload, we need to maintain its orientation. This is expressed by a virtual Cartesian-3 joint connected between the deck and the world frame.

After configuring the virtual articulations the operator can drag the camera with the mouse to scan the object. As described above, in this case we also integrated our TRACK hardware interface, which simplified motion specification. To save cost—total materials cost for TRACK was under $500 USD—we opted not to include haptic feedback in TRACK, potentially making it less applicable for constrained tasks. This example shows that constraint prioritization can mitigate the issue somewhat.

Figures 4, 5, and 6 give three additional examples: (1) a trench is inspected, with the support legs moving the deck to extend reachable trench length; (2) a rigidly mounted side-facing camera is made to pan and tilt with the motion both parametrized and constrained by virtual revolute joints; and (3) two limbs execute a pinching maneuver with the pinch distance and angles controlled by virtual prismatic and revolute joints. For the bimanual experiment the robot was partially supported by an overhead crane as simultaneously raising two limbs is not supported on the current hardware. The crane served as a safety-backup in the other experiments.

In our current implementation we design all virtual articulations and motions in simulation, export them as joint space waypoint sequences, typically generating a new waypoint whenever any joint moves at least 2°. We then check the sequences in a previously validated simulator and execute them as position-controlled trajectories on the hardware. We performed each of the four experiments at least twice, though such repeatability is actually a property of the hardware, not our interface system.

5 Limitations and Future Work

A next step for this work will be to perform measured usability experiments; for those to be meaningful we should implement a few additional critical usability features including snap-dragging [2] and undo. We will also implement a more self-documenting drag-and-drop UI for constructing virtual articulations. We envision measuring both the operator learning time for our system vs. existing systems at JPL, and also the time required to design a complex motion in each system. The comparison may be indirect as our system is higher-level than the others.

Our purely kinematic implementation applies only to fully-actuated cases; we are developing quasi-static extensions for some underactuated tasks. Some constraints, for example helical motion, cannot be modeled with the current set of joints in the system. Possible extensions could increase the set of representable constraints.
Fig. 4 ATHLETE performing a trenching motion (mixed real/virtual interface view inset).

Fig. 5 ATHLETE panning and tilting a fixed-mount camera with whole-robot motions.

Fig. 6 ATHLETE performing a bimanual pinching motion.
6 Summary

Our mixed real/virtual interface implements virtual articulations as a rapid graphical operator interface for coordinated manipulation and inspection motions in high-DoF articulated robots. This new method fills a gap between existing lower- and higher-level interfaces. It is topology-independent, supports scalable motion specification, and usefully handles both under- and over-constraint.

We used our interface to experimentally demonstrate four new classes of coordinated motion for NASA/JPL’s 36-DoF ATHLETE, all of which would have been difficult using prior methods, and we used constraint prioritization to combine our inexpensive direct manipulation device with virtual motion constraints.

Acknowledgements ATHLETE VRML model provided by RSVP team, NASA/JPL/Caltech. Work with ATHLETE hardware was carried out at the Jet Propulsion Laboratory, California Institute of Technology, under a contract with NASA and funded through the Director’s Research and Development Fund. Additional funding came from the NSF EFRI program.

References