FORK^{-1}S: Interactive Compliant Mechanisms with Parallel State Computation

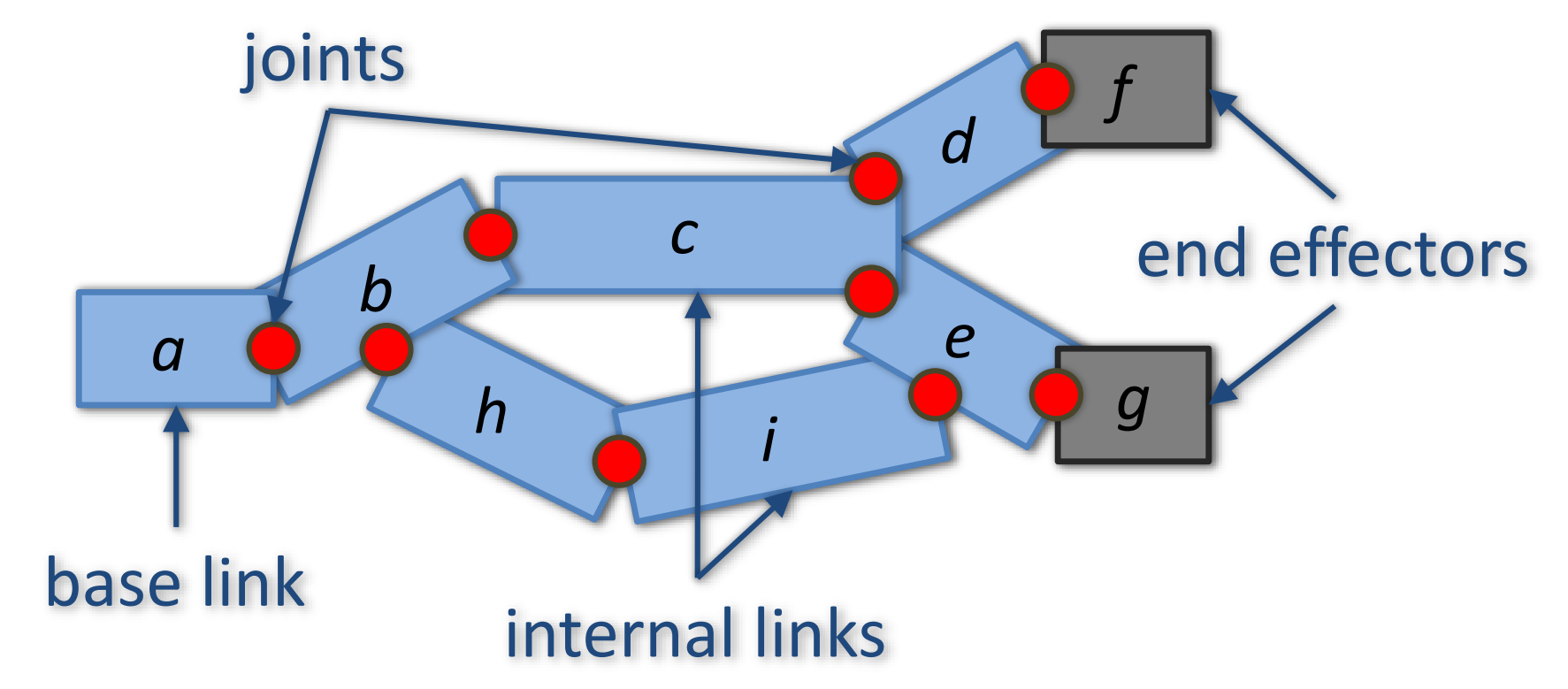
Sheldon Andrews[1], Marek Teichmann[2], and Paul G. Kry[1]

McGill University[1] CM Labs Simulations[2]

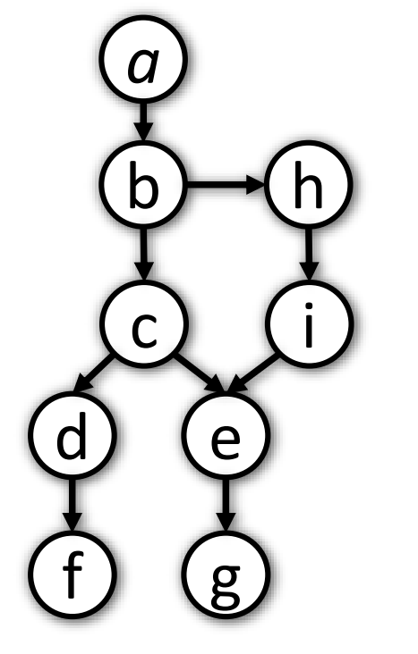
Introduction

We present a method for the simulation of compliant, articulated mechanisms using an approximate model that focuses on endpoint interaction. We approximate the mechanism's behaviour about a reference configuration, resulting in a first order reduced compliant system, or FORK^{-1}S [1]. Our approach is fast, and computation of the full structure's state is parallelized. The approach is suitable for articulate mechanisms using static proportional derivative (PD), such as those found in robotic simulation or physics based character animation. We demonstrate that simulations with our method can deal with kinematic chains with loops and non-uniform joint stiffness, and that it produces plausible effects due to stiffness, damping, and inertia.

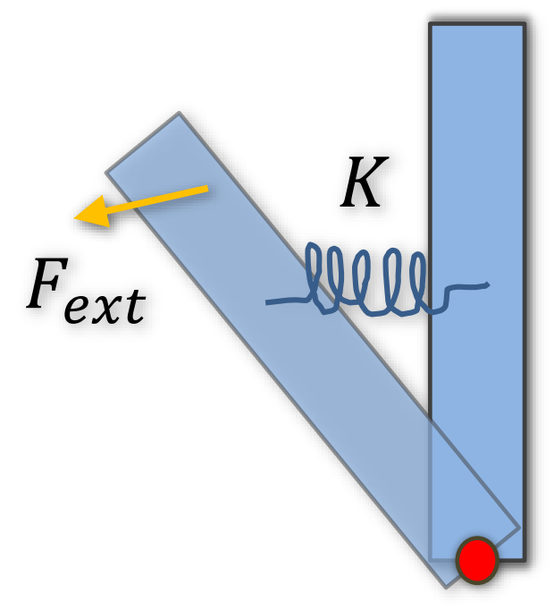
The FORK^{-1}S Model



A mechanism consists of a collection of rigid bodies with mass and inertia. Their relative motion is coupled due to joint constraints.



The joint topology of a mechanism may be represented as a directed acyclic graph (DAG). We use a recursive algorithm to traverse the graph and incrementally compute the effective mass, stiffness, and damping of an articulated mechanism.



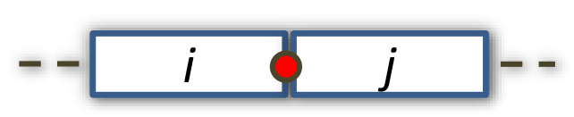
A wrench, F_{ext}, applied to this simple mechanism (right) causes the body to rotate about the joint. Assuming PD servo control, the equilibrium configuration, ξ, is determined by the linear system $K\xi = F_{ext}$, where K is the effective stiffness of a generalized spring.

The effective inverse stiffness, or compliance is approximated by a first order projection $K^{-1} = JK_\theta^{-1}J^T$, which includes the Jacobian J and the joint stiffness, K_θ. Similar projections may be used to compute approximations for the effective mass, M, and damping, D.

Building the Effective Matrices

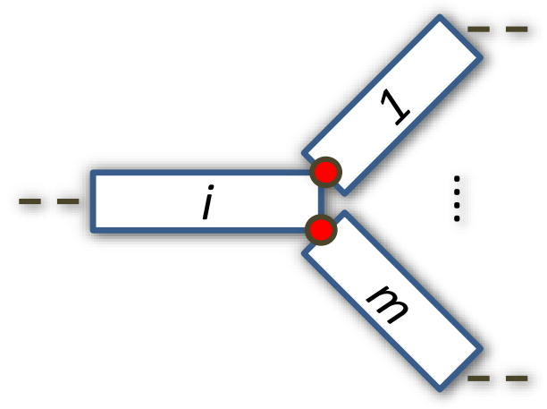
The effective mass, stiffness, and damping are built using a recursive algorithm as part of a pre-computation step. A forward pass starts at the base link and traverses the mechanism DAG towards the end effector nodes, incrementally computing the effective matrices of each link. This allows even complex mechanisms to be modeled. Three fundamental cases are considered:

Chaining



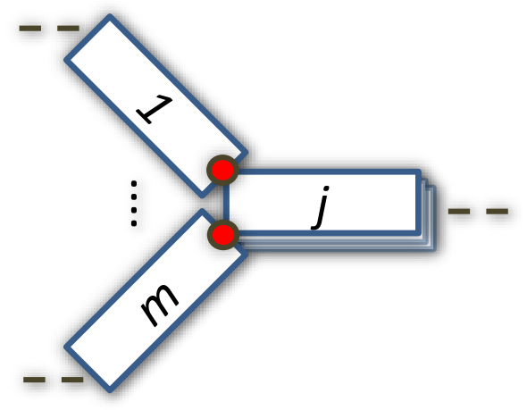
The effective compliance, K^{-1}, of a body is the sum of the effective compliance of the parent link and the compliance of the connecting joint.

Splitting



When two or more links share a common parent, a block matrix representing the effective compliance of the local system is constructed. The diagonal elements contain the compliance of each child link; the off-diagonal blocks provide the coupling between sibling links.

Merging



The effective stiffness is computed as a parallel combination of the effective stiffness of the parent links, including the coupled stiffness due to chains sharing a common ancestor.

Computing the Wrench Map

To update the internal links, we compute twists that explain their current state. The pre-computed wrench map, W, distributes wrenches at the end effectors to internal links in a plausible manner. The wrench map is incrementally constructed by a reverse traversal of the mechanism DAG. For end effectors E, the wrench at the ith internal link is computed as $F_i = \sum_{e\in E} {}^{i}W_e F_e$, where F_e is the external wrench at end effector e. The twist matching the static solution, ξ_i, of the internal link is directly computed by combining the wrench map with the effective compliance of the internal bodies. Special consideration is given to splitting and merging of the kinematic chain, since there is contributed motion among links that share a common ancestor. Therefore, the twist of an internal link is computed directly by

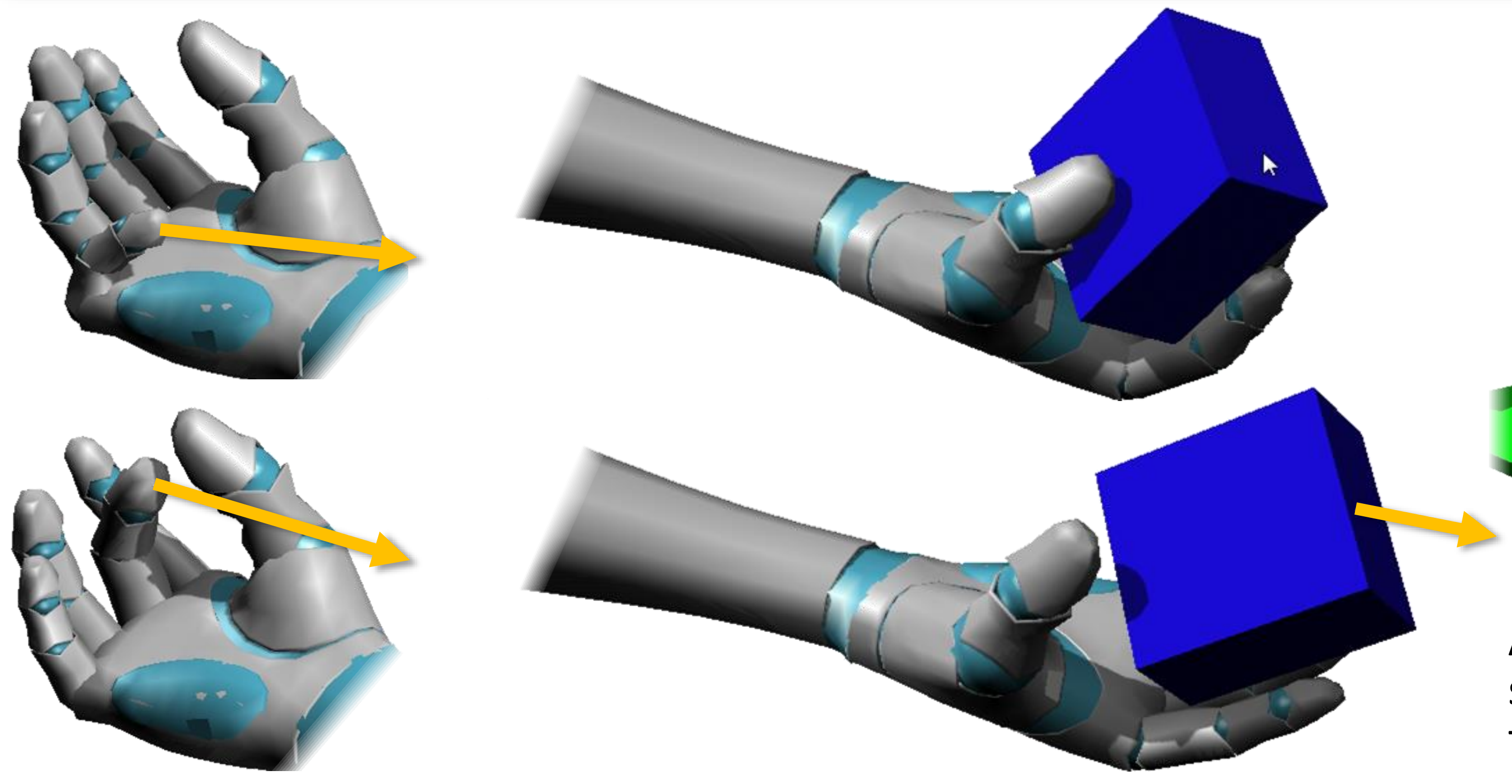
$$\xi_i = \sum_{e\in \mathrm{E}}\left(K_i^{-1}\ {}^{i}W_e + \sum_{a\in \mathrm{A}} {}^{i}_{a}Ad\ K_a^{-1}\left({}^{a}W_e - {}^{i}_{a}Ad^T\ {}^{i}W_e\right)\right) F_e\,,$$

where A is the set of ancestors of the internal link and the adjoint matrix ${}^{i}_{j}Ad$ maps twists from coordinate frame j to frame i.

Dynamical Simulation

The acceleration of the end effectors, $\dot{\phi}$, due to external wrenches is solved directly by $\dot{\phi} = (M + h^2K + hD)^{-1}\,F_{ext}$.

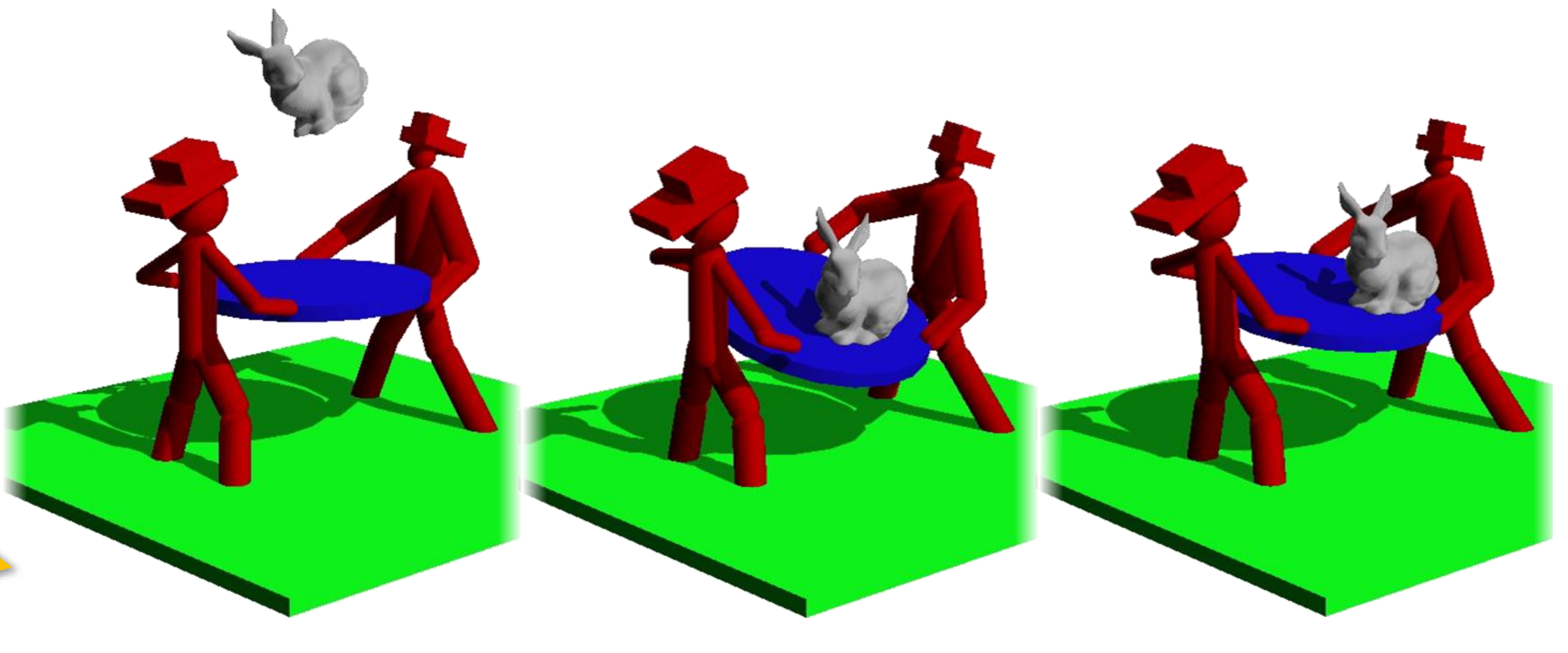
Results



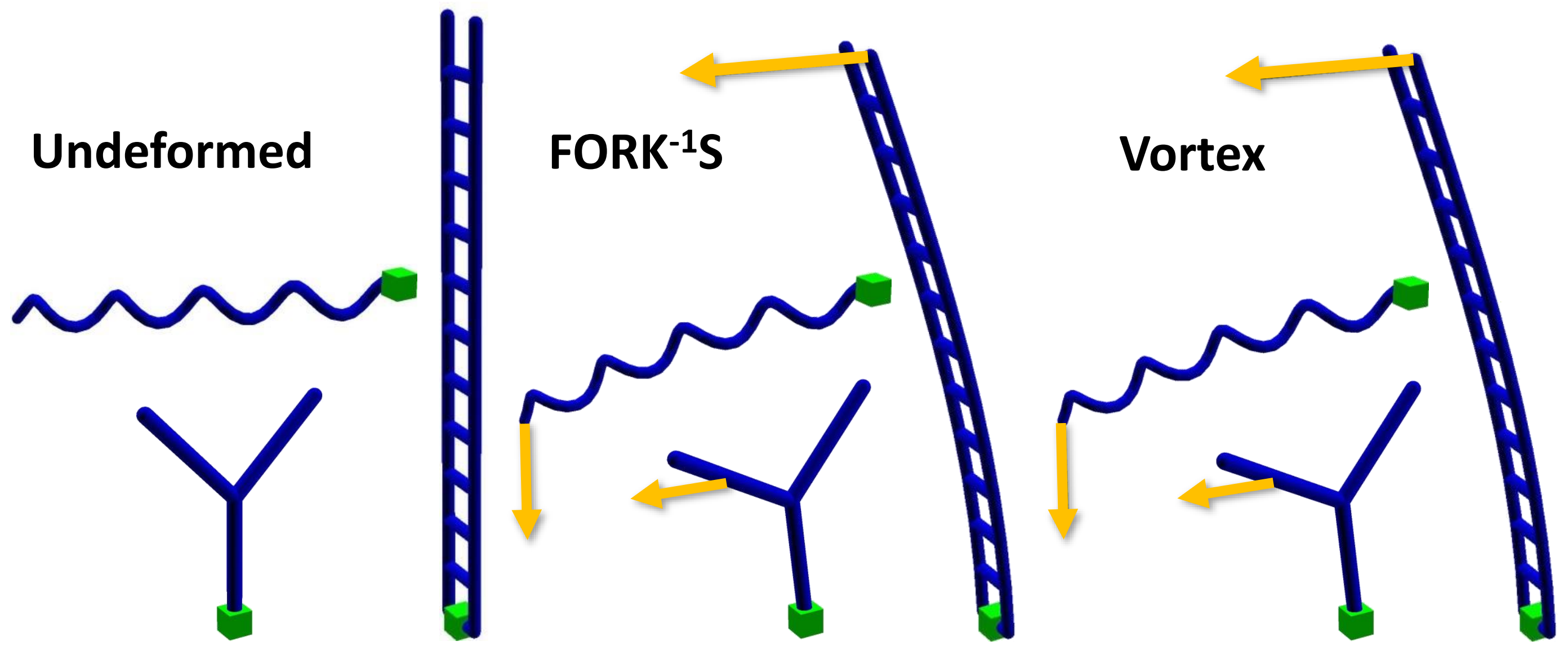
Dynamics simulation is used to update only the fingertips of this robot arm, but the behaviour of the kinematic chain is physically plausible. A yellow arrow is used to visualize an applied external force.

Example	# links	Vortex	FORK^{-1}S		
			1 thread	4 threads	8 threads
Helix	50	240	20	12	15
	100	470	32	16	21
	400	2150	112	84	76
Ladder	48	334	22	14	18
Robot Arm	20	121	24	18	21

Computation time (μs) per simulation frame for various examples. The largest observed speedup was 28x faster compared to a commercial rigid body physics engine, Vortex [2].



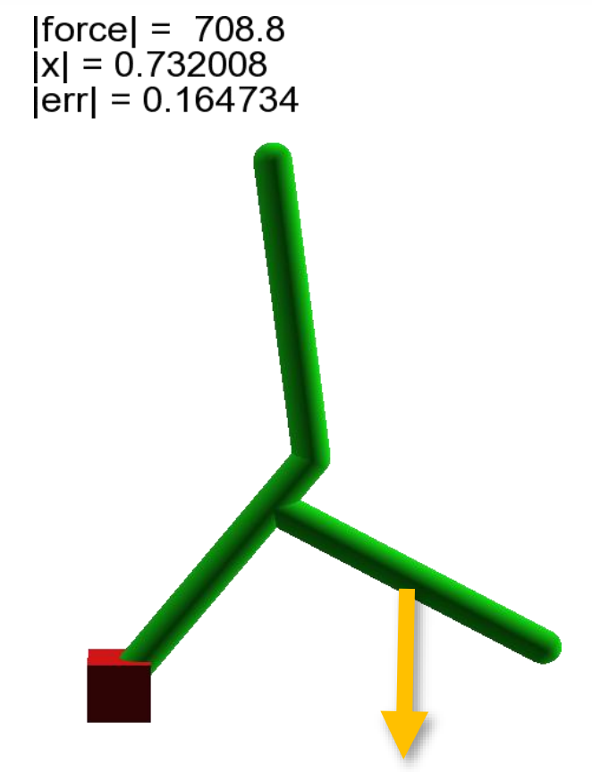
A pair of "whacky" firemen catching a bunny. Note that there are several loop structures in the kinematic chain, and only the bodies of the trampoline and the bunny are simulated using dynamics.

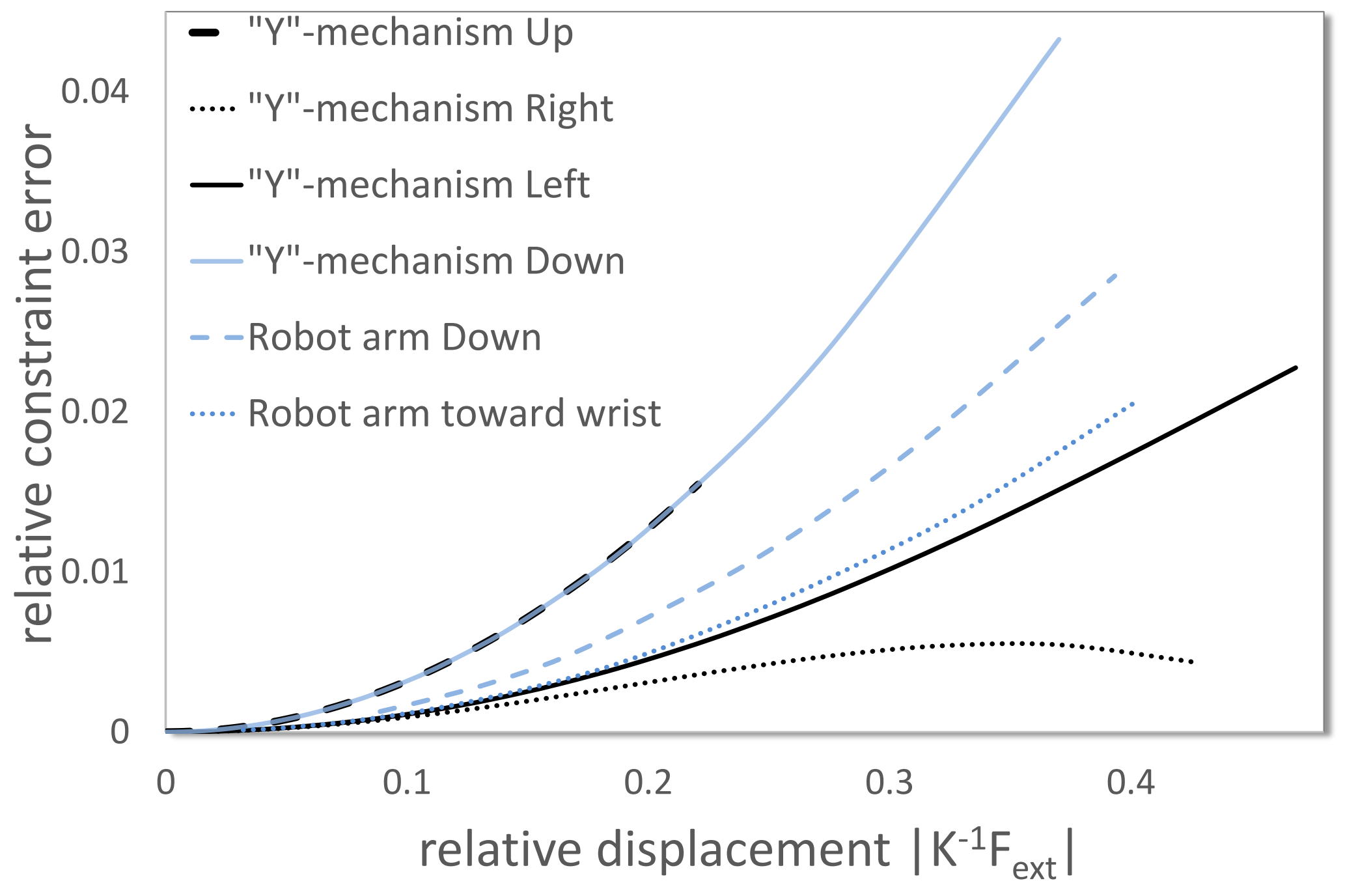


Comparing the static equilibrium configurations for various mechanisms. The differences between a FORK^{-1}S simulation and a Vortex multi-body dynamical simulation are visually imperceptible.

Limitations

Constraint violation errors can occur. For large displacements of the end effector, our first order model, which is a linearization about a single point in the mechanism's configuration space, no longer well approximates the mechanism's behaviour.





The relative constraint error is computed as the sum of the constraint violation across all joints, which is then scaled by the bounding sphere radius of the mechanism. The relative error remains low, even for significant displacements of the end effector. The rate at which the error increases depends on the direction of the applied force.

Conclusions

FORK^{-1}S provides an important new approach among a large spectrum of techniques for the creation of interactive and immersive virtual environments. First-order reduced models of compliant mechanisms provide a fast alternative for virtual human and robot simulation. By focusing on the end effectors, the simulation only solves a small dense system while the full state of the non-reduced mechanism can be computed in parallel.

Video Results

Video results can be seen on the project page: http://www.cs.mcgill.ca/~sandre17/forks

References

[1] S. Andrews, M. Teichmann, and P.G. Kry. (2014) *FORK^{-1}S: Interactive compliant mechanisms with parallel state computation.* The ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games.

[2] CM Labs Simulations. (2012). Vortex SDK version 5.2.