Simulation-Based Tools For Evaluating Underactuated Hand Designs

Daniel M. Aukes and Mark R. Cutkosky

***Abstract*— This paper presents a tool aimed at the design of compliant, under-actuated hands. The particular motivation is hands that will be used for an underwater robot to grasp a variety of objects, some of which may be delicate or slippery. The focus of the analysis is the problem of object acquisition. In comparison to many prior grasp analysis tools, the tool presented here models the dynamics of a hand, including actuation mechanisms, compliance and friction in an efficient formulation that permits one to evaluate variations in such quantities as phalange length, finger spacing, transmission ratios, and torsional joint stiffness when comparing hand designs. The analysis is demonstrated for a quasi-static object acquisition problem and leads to the computation of a vector space of three dimensional regions for which the hand will tend to center and stably grasp a compact object.**

I. Introduction

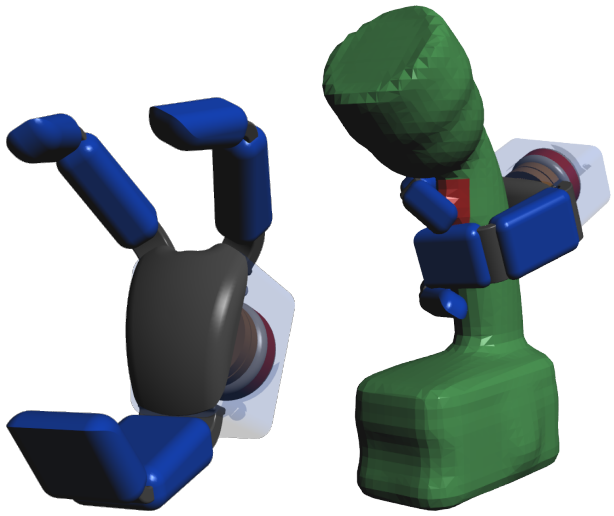
Over the last three decades, many grasp analysis approaches have been developed, ranging from kinematic studies[1] to evaluations of grasp synergies[2][3] and grasp force and contact location evaluations[4][5].

However, comparatively few analyses have focused on compliant and under-actuated hands, which have recently enjoyed renewed popularity as a light, robust and versatile grasping solution for mobile manipulation applications [6][7][8][9].

One reason for the discrepancy is the scarcity of dynamics packages that are well suited for addressing the specific challenges associated with under-actuated grasping, including frequent formation and breaking of contacts, friction and sliding, and compliance in the actuation system. Notable recent efforts to address the design of under-actuated hands include Dollar et al. [10], who map regions of stable grasps for a compliant planar hand with the goal of optimizing key design parameters, such as resting angle and joint stiffness. Balasubramanian et al. study how different transmissions and control modes affect grasping ability for underactuated hands in [11][12]. Other analyses have utilized the GraspIt! dynamics engine [13] combined with a quasi-static quadratic program (QP) solver to find key force information across a variety of grasps. As noted in [14], the QP solver used for computing static forces in under-actuated hands remains a work in progress. Kragten et al. [15] discuss popular hand performance metrics used across the literature and present a new metric using work performed on grasped objects with frictionless, multi-phalanx underactuated hands. Aukes et al. [6][7] also use the concept of pullout forces and work in analyzing two different designs. In other work, Hammond et al. [16] have used GraspIt!'s native grasp planner to optimize a compliant, four-finger under-actuated hand.

The authors are with the Department of Mechanical Engineering, Stanford University, Stanford, CA.

D. Aukes danaukes@stanford.edu



(a) (b)

Fig. 1: Three-fingered under-actuated hand (a), grasping a compact object (b).

Although under-actuated hands and grasps pose particular challenges with respect to modeling friction, compliance and varying contacts, these problems are also encountered more generally in three-dimensional multi-body dynamics simulation, for which a variety of packages have been developed. However, the choices made for each dynamics engine may limit usefulness when applied to under-actuated grasping. Some of these design choices include time-stepping methods, integration methods, equation formulation, contact detection and equation solving methods, as well as more practical concerns such stability, speed and the accessibility of contact data for monitoring. Dicussions of dynamic simulation approaches and trade-offs include [17][18][19].

In this paper we discuss the elements of simulation relevant for evaluating under-actuated hand performance, introduce methods for generating data from a selected simulator, and describe the methods used to investigate the impact of changing specific design parameters on a selected performance metric. We present some initial findings and discuss future directions in which such analyses might lead.

II. Dynamics Engines

Much work has also been done on formulating expressions for simultaneously solving dynamics equations with one-sided inequalities such as coulomb friction and

Dynamics Engine	Time Stepping Method	Contact Formulation(solver)	coulomb friction	Contact Detection	Reduced Coordinates	Rigid Constraints	Interface
ODE	Fixed	LCP(Pivoting)	N	Y	N	Y/N	C++/Python
GraspIT!	Fixed	LCP(Lemke)	Y	Coin	N	Y	C++/GUI
Moby	Variable/Fixed	QP	Y	CCD(custom)	Y	Y	C++
SimPack	Variable/Fixed	Penalty	Y	Y(several methods)	Y	Y	GUI/text

TABLE I: Comparison of dynamics engines

XML File	C Interface	Python Simulation Framework
Define bodies	Collect state data every timestep	Define simulation parameters
Define kinematics	Collect collision event data	Generate XML File
Define geometries	Connect python controller to library	Spawn parallel simulations
Define contact geometries and parameters		Log state and collision data
Select integration method		Control Joints

TABLE II: The three parts of the simulation framework.

joint limits. Using Linear Complementarity Problem (LCP) solvers[20][21], simulators have been able to move away from the problems traditionally found with penalty-based collision methods. Other quadratic programming (QP) methods have been used to similar success [13],[22]. Because of their general-purpose ability to simultaneously solve both dynamics and one-sided contact and limit problems, and the increasing performance of today's computers, their capabilities can be utilized for a wide range of static and dynamic analysis problems found in grasping.

There are many items on the wishlist for an ideal dynamics package. When it comes to improving simulation speed, variable time-step integration is desirable but not often found in dynamics engines. Not only does it reduce the number of parameters that can influence simulation stability, it allows more automated simulation of mechanisms with different impedences. Reduced coordinate descriptions are also preferred, because maximal coordinate representations, while simple in their implementation, can incur unneccessary computing loads. This slowdown can become apparent when many bodies are added to the system, as in the case of 3 or 4-finger, multi-phalanx hands. Due to their structure, dynamics calculations on articulated robots can also benefit from recursive algorithms[17]. Simulation stability is always a concern, especially when dealing with compliant elements that may introduce oscillations. Energy-neutral calculations are preferred when it comes to friction approximations and restitution calculations, and while difficult to compare without testing, they should be considered when searching for a suitable dynamics engine. More mundane details of each dynamics engine cannot be discounted either, such as the ease of retrieving state and contact data, setting up and defining passive joints, and passing simulation parameters to the engine. A comparison of some common tools can be seen in table I. One common dynamics software package is ODE, which has found its way into many popular robotics simulation software packages such as Gazebo and OpenRAVE. While popular, it suffers from some disadvantages when considered for this application. Its manual suggests using a constraint force mixing parameter to add soft constraint forces to each joint rather than rigid constraints. This can lead to instability when the stiffness of compliant joints approaches the stiffness of the joint constraints, sometimes causing the entire assembly to behave in ways not kinematically possible. ODE uses fixed-step integration and a maximal-coordinate desciption to describe the state of each body. Joint limits cannot be defined natively, and must be added to the system as penalty forces instead of being included in its iterative LCP formulation. Finally, contact forces are not readily available at each timestep, making recording and understanding the interactions between bodies in the simulation rather difficult.

When comparing dynamics engines, GraspIt! must also be considered, since at its core it also contains a multi-body

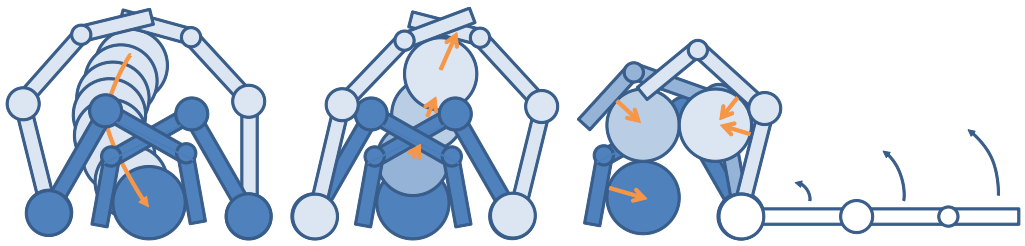


Fig. 2: Methods for estimating grasp acquisition volume. (a) indicates an object being drawn into the palm from an arbitrary starting position, (b) shows the effect of pulling an object out of a stable grasp in a certain direction, and (c) represents the changing contact forces due to moving a fixed object around the finger workspace.

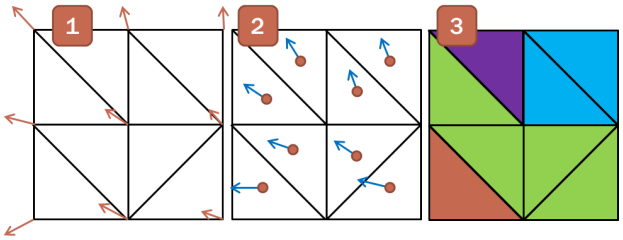
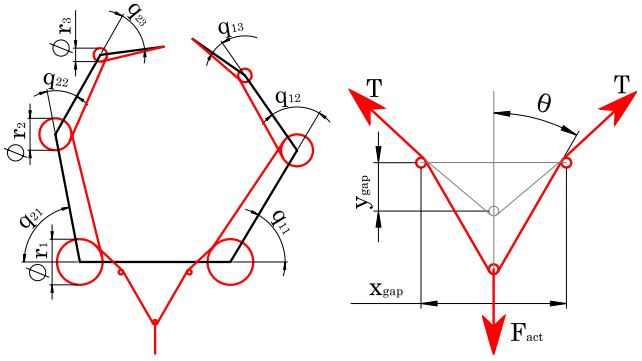


Fig. 3: The three steps involved in calulating the region of acquisition: 2) divide the gridded force-field, 3) determine each region's connection using the average force at the centroid, 4) collect connected regions.



(a) (b)

Fig. 4: The kinematics of the planar coupled hand. The motion of any joint causes a change in the total tendon length, and thus the gear ratio applied from a linear actuator. Pulley routing is shown in red, with actuation supplied by a linear actuator pulling with force F_{act}.

dynamics engine. Like ODE, the simulator uses a maximal-coordinate representation for each body and integrates using a fixed time-step. It includes an implementation of the LCP formulation first described in [20] and implemented in [21].

Moby is an open-source dynamics package with a plugin interface to its C++ libraries that gives users the flexibility to pick and choose among several integrators, QP solvers, and dynamics calculation methods[23]. ABA and CRBA algorithms are available for calculating dynamics on branching articulated robots[17], variable-step integration is available via the ODEPACK library, and reduced-coordinate descriptions can be used for articulated robots.

Other commercial dynamics packages are available, but less is known (and published) about their inner workings. Common, well known packages are ADAMS and SimPack.

III. Method Selection

Other research, e.g. [15], has evaluated performance using potential fields and force/position profiles as an evaluation tool. In this study Moby is used to determine the volume in which an object can be acquired from the field of resultant forces in the hand's workspace. Given Moby's flexibility, several testing scenarios seem feasible:

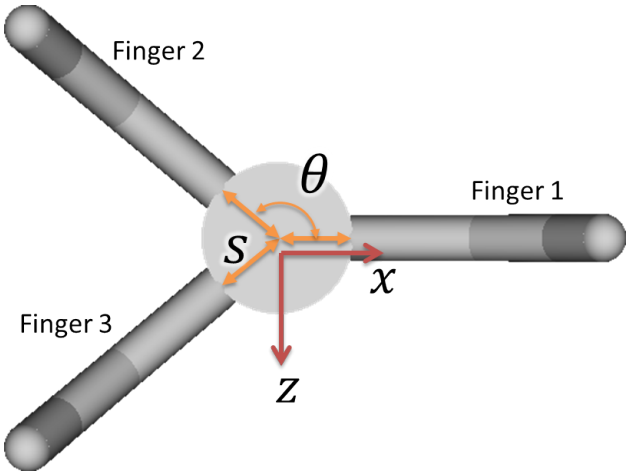


Fig. 6: Layout of the three-fingered hand. θ represents the orientation of the hand and s represents spacing of the base joint from the center of the palm.

variable	value	unit
finger length	125	mm
finger diameter	10	mm
link lengths	57, 40, 28	mm
pulley radii	12.5, 8.75,3.5	mm
spring stiffness	.1	N-m/rad
spring preload	0	N-m
upper joint limits	120,95,95	deg
lower joint limits	0,-20,0	deg
actuator force	40	N
object radii	20, 30, 40, 50	mm

TABLE III: Parameters used in hand analysis

1) **Direct Test:** Start an object from many initial locations within the hand's workspace to determine the acquisition region (Fig. 2a).
2) **Prescribed Force, Calculated Position:** From a stable grasp at the palm, slowly pull an object out with increasing force under quasi-static conditions to create a vector field of force/position relationships from the collected contact data (Fig. 2b).
3) **Prescribed Position, Calculated Force:** Fix an object in space and let the mechanism settle to static equilibrium, and record the resultant force on the object (Fig. 2c). Repeat this simulation for many fixed positions throughout the workspace to compile a similar force/position map.

Using method 1, the hand can be fully defined and tested in near-real-world conditions. The friction conditions can be set to those that might be expected, for example, and specific objects can be tested which may be expected for a certain hand. While it is a very direct, simple method of determining the volume of graspable objects, this method seems intractable. Due to the specificity of each test, generic conclusions cannot be easily made about the relationship between a design parameter and the hand's general ability to grasp without running thousands of simulations across a wide range of objects types, friction cases, and loading conditions. In addition, performing full dynamic simulations for a single

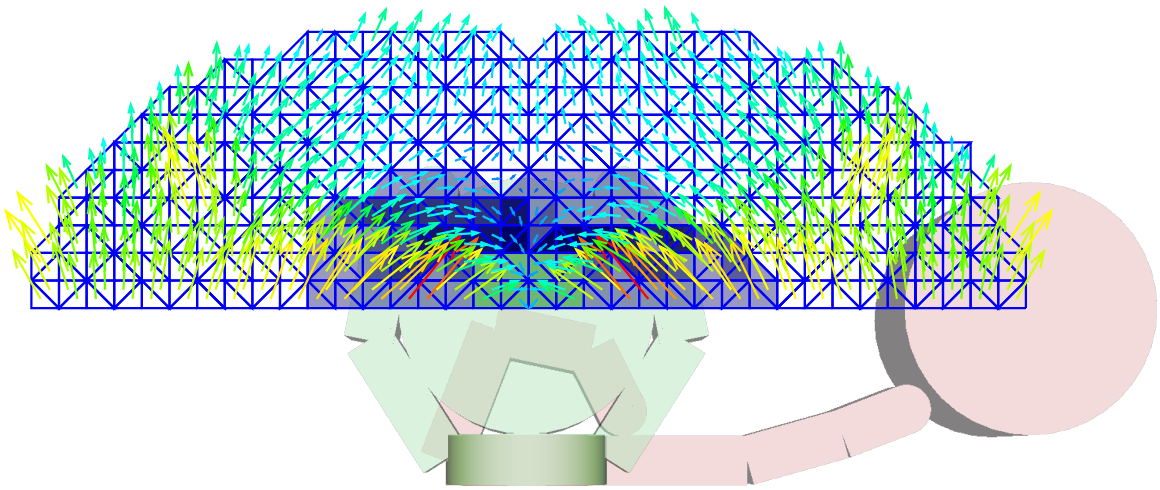
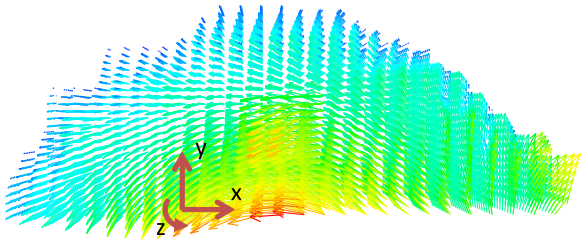
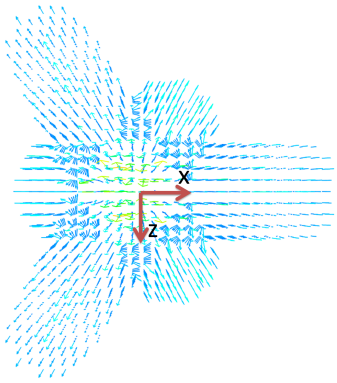


Fig. 5: Force field for a planar, coupled hand. The blue lines show the grid of triangles produced by a delaunay triangulation of the area, the arrows represent the average forces for each area at its centroid, the blue region represents triangles whose forces point toward the palm, and the green region shows the goal region located at the palm.



(a)



(b)

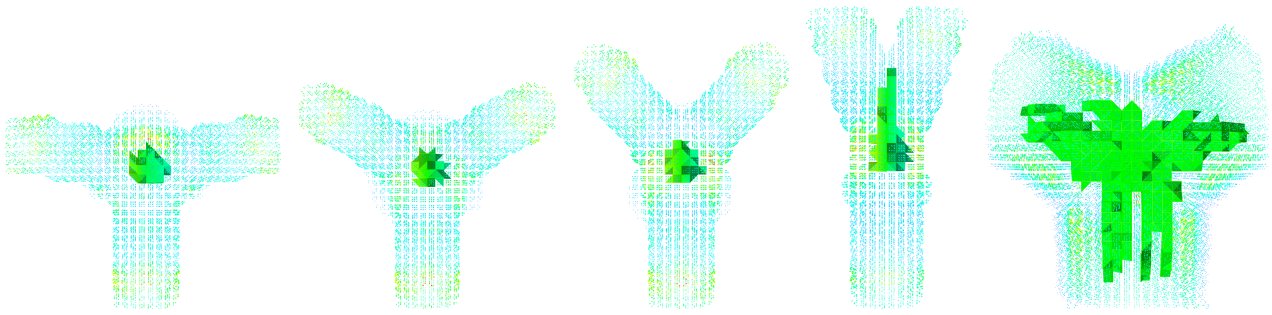
Fig. 7: Fig. (a) shows the three-dimensional field of resultant forces on an 50mm object over the workspace of finger 1. (b) shows the field on a hand design with $\theta = 120$ degrees and $s = 30$ mm.

object/mechanism system at many points throughout the hand's workspace is a time-consuming method of generating such a volume.

Like method 1, many different friction, object geometry, and kinematic conditions can be tested with method 2, with some key advantages over the former. By slowly changing an external disturbance force on the object, allowing the system to come to rest, and recording the state of hand and object, a quasi-static force/displacement relationship can be obtained between the hand and object. Unlike method 1, since the recorded force/position relationships are collected when the system is near-static, the hand's kinematic and force-generating abilities influence grasping performance more than object or finger mass. Because the object is pulled from an initial stable equilibrium in the palm, it is impossible to map the force/position relationship outside of that stability envelope, and it may be difficult without a prohibitively small force increment to precisely find the boundary at all locations. Yet as with method 1, simulation must be performed with a full hand model, and simulation speed will decrease as the number of phalanges and contacts increase.

Instead of finding the position as a function of the force, the third method finds the resultant force on an object as a function of its position in the mechanism workspace. The object is placed in a set of (gridded) positions, and the resultant force on the object is calculated as a function of the tendon tension. Because the object is fixed in space, the mechanism/object interaction can be tested even in unstable regions of the workspace, which is advantageous because independent subsections of a hand mechanism can then be investigated independently and then superimposed when possible, greatly increasing simulation speed.

Due to the path-dependent nature of friction, including it using method 3 makes less sense because the finger is brought into contact with the fixed object for each position/force trial. Hence, the static state generated by such a test would not necessarily represent a realistic grasping posture. A path-independent, frictionless test would be useful, though, because the force/position relationships are just as useful in object acquisition as object retention analyses. Zero-friction conditions can also be considered a worst-case scenario for grasping, as the hand's kinematics alone must be able to attain static equilibrium with the object using only normal contact forces.



(a) (b) (c) (d) (e)

Fig. 8: The effect of finger orientation on grasping volume for a 40mm sphere is shown. As the orientation of the fingers varies between 100(a), 120(b), 140(c) and 160(d) degrees, the measure of space where an object may be acquired changes both in shape and volume, measured in the number of tetrahedra connected to the palm. Changing the shape of the phalanges from 20mm-diameter cylinders (a)-(d) to 60mm-wide boxes (e) increases the acquisition volume as well.

Despite its limitations, method 3 has been selected for comparing performance across hand designs because of its potential for efficiently calculating worst-case hand forces.

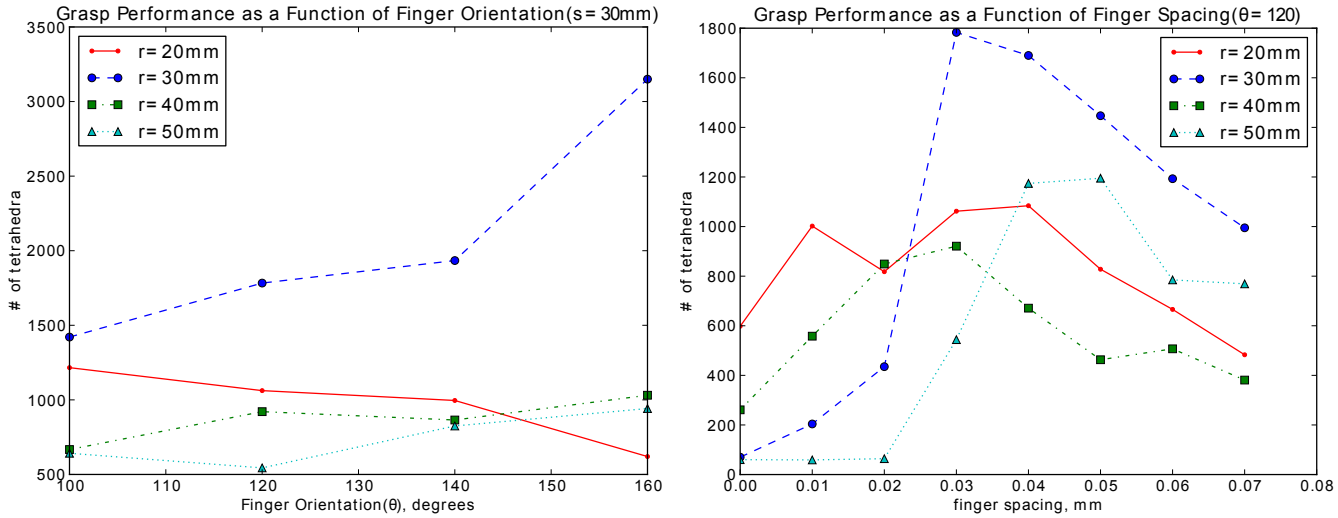
IV. Calculating Hand Performance

There are several steps requried to calculate an acquisition region between a hand and object. For devices composed of independent mechanisms, such as hands with identical fingers where sub-mechanisms do not interact with each other directly, it can be useful to analyze each sub-mechanism independently. While there are several more steps and restrictions in doing this, splitting the mechanism can dramatically improve simulation speed while still giving valuable design information. The summary of steps is as follows, which can be repeated across many objects and designs:

1) Separate Hand into independent units. (optional)
2) Determine static forces contact forces on a mechanism / sub-mechanism using method 3 outlined in III.
3) Use the principle of superposition to reconstruct the force field for the full mechanism. (optional)
4) Determine acquisition region from force-field and calculate volume.

A. Calculating Static Forces

Moby's simulation environment is defined with a custom XML file, which allows users to create sets of independent and articulated rigid bodies, define joint properties, attach contact geometry and set collision parameters. It is also used to specify the integration method and dynamics algorithm. Among these choices, the CRBA method with variable-step integration has been found to be efficient and useful. A custom interface developed for Moby allows joint torque control and impact logging between the robot and a grasped



(a) (b)

Fig. 9: Grasp Performance as a function of finger orientation and spacing. (a) shows the changing grasp volume as a function of finger orientation, θ, for several different size spheres. (b) shows the impact of varying finger spacing, s.

object, with control and logging functions written in Python. Joint torques are calculated by applying spring moments according to each joint's position at each timestep and adding actuator torques scaled by the effective transmission ratio.

For each position in a grid of the finger workspace, a separate simulation is created with the grasped object fixed in the specified x, y, z position. Tendon forces are then applied to each joint, and the finger begins to contact the object. To facilitate settling, a small damping term is applied to each joint. Once the system has settled, contact events are recorded between the hand and object over a certain time window, and the tendon force is increased by a small amount. The process is then repeated up to the specified force limits in the simulation. Moby calculates impacts as discrete events in time, determining the impulse which solves the LCP for each discrete contact event. Thus for each recording window, any impulses that occur are logged and averaged over the recording time, producing a static contact force. The resultant force on the object is calculated as the sum of all contact forces acting through the center of the sphere, and resultant moment as the sum of all individual moments due to contact, summed about the center of mass. Note that the resulting moment for circles (2d analysis) and spheres (3d analysis) is $\mathbf{0}$ in the absence of tangential forces due to friction because all normal contact forces act through the center of mass.

B. Acquisition Region Calculation

To evaluate the region of space where the object can be acquired by our hand, the object's motion through this field must be determined. Since these data are generated while the system is close to static equilibrium, quasi-static object motion is assumed. In this case it is not necessary to find the second-order integral of a mass traveling through the field, but is sufficient to simply consider first order paths. Integrating the first order differential equations is one way to determine such a trajectory, but the additional interpolation and calculation costs are quite high over many points. Instead, an approximation of the volume is measured by evaluating the elements of a Delaunay triangulation, where each point in the grid becomes a vertex for an element in the triangulation (tetrahedra in 3-D, triangles in 2-D). Instead of evaluating the full path the object travels, it becomes only necessary to determine the connectedness of each element. To accomplish this, the direction of the resultant force at each element's centroid must be calculated, and the neighbor which it points to becomes the next element in the path. A matrix $\mathbf{A}$ is created for the whole triangulation, where element $a_{i,j} = 1$ if element i points to element j. From matrix $\mathbf{A}$, a directed graph is obtained in the form of $\mathbf{B}$, where $\mathbf{B}_0 = \mathbf{A}$, and

$$\mathbf{B}_{k+1} = \mathbf{A}\mathbf{B}_k + \mathbf{A}. \tag{1}$$

When, for iteration $n + 1$, no more elements $b_{i,j}$ become nonzero, iteration stops. The resulting matrix, $\mathbf{B}_n$, describes how force regions are connected together. Fig. 3 outlines these steps.

Moby has been used to understand the aquisition ability of two different tendon-driven, compliant hands: a 2-finger, 3-phalanx hand with a coupled transmission, and a 3-finger, 3-phalanx hand with varying finger layout geometries and surface shapes.

C. Analysis I

The first analysis is performed here on a planar, tendon-driven finger with joint compliance at every joint. The drive tendon is coupled between both fingers by a pulley attached to a linear drive motor. Referring to Fig. 4, the total, constant, available length of the tendon can be calculated as

$$L_{total} = L_{gap}(t) + c + \sum r_j \left(q_{ij}(t) - ll_j\right), \tag{2}$$

where r_j is the radius of joint j, $q_{ij}(t)$ is the rotation of joint j of finger i, and ll_j is the angle of the lower stop of joint j, c is a constant term which accounts for other unchanging gaps between pulleys. $L_{gap}(t)$ represents the amount of tendon spanning the two fingers, with its initial length calculated as $\sqrt{x_{gap}^2 + 4y_{gap}^2}$ if small pulleys are assumed in Fig. 4b. Because L_{total} is constant, L_{gap} is therefore dependent on each joint variable. The tension of the tendon is related to the force in the actuator by the relation which solves the y-component of the free body diagram at the actuator connection, with

$$F_{act} = 2T\cos\theta \tag{3}$$

$$\text{and } \theta = \sin^{-1}\left(x_{gap}/L_{gap}\left(t\right)\right). \tag{4}$$

The results of the analysis can be seen in Fig. 5.

D. Analysis II

The goal of this study is to find an optimal design for finger orientation and spacing (θ and s, respectively) in a three-dimensional design, as shown in Fig. 6. Other design parameters have remained constant in this study, such as finger length, joint stiffness, and tendon travel. Table III summarizes the design parameters used in our analysis.

To improve the performance of the three-dimensional simulation, a single finger was simulated and the results superimposed, as seen in Fig. 7. While this results in significant speed improvements due to the reduction in the number of bodies, it requires several assumptions. First, only spherical objects are evaluated, because the superposition of three fingers' force data can only be done if the results are invariant to obect orientation. Another assumption is that interference between fingers does not occur. This can be justified either by evaluating only sufficiently large objects or evaluating designs where fingers are located so as to not collide. These assumptions can be relaxed using a full hand simulation at the expense of much slower computation.

Once simulations have been performed over a grid of the 3-D finger/object workspace, the resultant object forces are superimposed to determine hand forces. In this example study, hand designs have been calculated for orientations $\theta = \{100, 120, 140, 160\}$ deg, and spacings $s = \{0, 10, 20, 30, 40, 50, 60, 70\}$ mm. A transformation matrix is

created for each finger as a function of its rotation about the y axis (the axis that exits the center of the palm) and the base joint's distance from the center. A new hand grid is created, finger data are reinterpolated for each finger position, and the results are added. The field of forces over a particular full hand design can be seen in Fig. 7b.

V. Discussion

Fig. 9 shows the results of the finger placement analysis performed over four different spheres with $r = \{20, 30, 40, 50\}$mm. While finger spacings between 30 and 50mm seem to produce the best overall grasp performance over our range of objects, finger orientation does not have a strong impact for most of the objects on the grasp volume. The shape of the volumes in Fig. 8 changes across the tested orientations for the 40mm object, indicating that grasp performance may have as much to do with the shape of the space as its absolute volume. In addition, the effect of changing phalanx surface geometry can be seen in the comparison between 8b and 8e. In otherwise similar simulations, the cylindrical phalanx geometry seen in Fig. 6 is replaced with a box 60mm wide by 10mm deep. The resulting acquisition volume is much larger than with cylindrical phalanges, underscoring the benefit of being able to test surface geometries.

One noticeable artifact of applying gridded tetrahedral meshes to a symmetric hand design can also be seen in Fig. 8. While the calculated volume is nearly symmetric, its lack of perfect symmetry between symmetrically rotated fingers indicates the effect of the mesh itself on the volume calculation. An improvement to this method would be to calculate connected volumes from randomly distributed points throughout the workspace, making the results of the analysis less orientation-dependent.

VI. Conclusions and Future Work

We present this analysis as a first step toward what we hope will be a simulation framework for analyzing hand performance. Future work will include experiments to confirm that calculated acquisition regions are achievable, along with comparisons with the other simulation methods described in Section III. With the ability to apply both joint and contact friction using the same tools, we hope future studies will continue to play a role as we move through the design phase.

VII. Acknowledgments

The authors thank Barrett Heyneman for his insights on many aspects of this paper.

References

[1] M. Mason and J. S. Jr, *Robot hands and the mechanics of manipulation*. Cambridge, MA: MIT press, 1985.

[2] A. Bicchi, M. Gabiccini, and M. Santello, "Modelling natural and artificial hands with synergies.," *Philosophical transactions of the Royal Society of London. Series B, Biological sciences*, vol. 366, pp. 3153–61, Nov. 2011.

[3] M. Gabiccini, a. Bicchi, D. Prattichizzo, and M. Malvezzi, "On the role of hand synergies in the optimal choice of grasping forces," *Autonomous Robots*, vol. 31, pp. 235–252, July 2011.

[4] A. Miller and P. Allen, "GraspIt!," *IEEE Robotics & Automation Magazine*, vol. 11, pp. 110–122, Dec. 2004.

[5] M. Ciocarlie, C. Goldfeder, and P. Allen, "Dimensionality reduction for hand-independent dexterous robotic grasping," in *International Conference on Intelligent Robots and Systems*, Citeseer, 2007.

[6] D. Aukes, B. Heyneman, V. Duchaine, and M. R. Cutkosky, "Varying spring preloads to select grasp strategies in an adaptive hand," in *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1373–1379, IEEE, Sept. 2011.

[7] D. Aukes, S. Kim, P. Garcia, A. Edsinger, and M. R. Cutkosky, "Selectively Compliant Underactuated Hand for Mobile Manipulation," *2012 IEEE International Conference on Robotics and Automation*, 2012.

[8] M. Carrozza, C. Suppo, and F. Sebastiani, "The SPRING hand: development of a self-adaptive prosthesis for restoring natural grasping," *Autonomous ...*, pp. 125–141, 2004.

[9] C. Gosselin, F. Pelletier, and T. Laliberte, "An anthropomorphic underactuated robotic hand with 15 dofs and a single actuator," in *2008 IEEE International Conference on Robotics and Automation*, pp. 749–754, IEEE, May 2008.

[10] A. M. Dollar and R. D. Howe, "Towards grasping in unstructured environments: Grasper compliance and configuration optimization," *Advanced Robotics, special issue on Compliant Motion*, vol. vol19, no. 5, pp. pp523–544, 2005.

[11] R. Balasubramanian, "External disturbances and coupling mechanisms in underactuated hands," in *Proceedings of ASME IDETC /CIE*, 2010.

[12] R. Balasubramanian, J. T. Belter, and A. M. Dollar, "Disturbance Response of Two-Link Underactuated Serial-Link Chains," *Journal of Mechanisms and Robotics*, vol. 4, no. 2, p. 021013, 2012.

[13] M. Ciocarlie and P. Allen, "A design and analysis tool for underactuated compliant hands," *Proceedings of the 2009 IEEE/RSJ*, pp. 5234–5239, 2009.

[14] "GraspIt! User Manual." `http://downloads.sourceforge.net/project/graspit/manuals/graspit-2.2_manual.pdf`.

[15] G. A. Kragten and J. L. Herder, "The ability of underactuated hands to grasp and hold objects," *Mechanism and Machine Theory*, vol. 45, pp. 408–425, Mar. 2010.

[16] F. L. Hammond, J. Weisz, A. A. de la Llera Kurth, P. K. Allen, and R. D. Howe, "Towards a design optimization method for reducing the mechanical complexity of underactuated robotic hands," in *2012 IEEE International Conference on Robotics and Automation*, pp. 2843–2850, IEEE, May 2012.

[17] R. Featherstone and D. Orin, "Robot dynamics: equations and algorithms," in *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No.00CH37065)*, vol. 1, pp. 826–834, IEEE, 2000.

[18] R. Gillespie and J. Colgate, "A survey of multibody dynamics for virtual environments," in *ASME Int. Mech Eng Congr Exp*, (Dallas, TX), 1997.

[19] A. Boeing and T. Bräunl, "Evaluation of real-time physics simulation systems," in *Proceedings of the 5th international conference on Computer graphics and interactive techniques in Australia and Southeast Asia - GRAPHITE '07*, vol. 1, (New York, New York, USA), p. 281, ACM Press, 2007.

[20] M. Anitescu and F. Potra, "Formulating dynamic multi-rigid-body contact problems with friction as solvable linear complementarity problems," *Nonlinear Dynamics*, vol. 14, no. 3, pp. 231–247, 1997.

[21] A. Miller and H. Christensen, "Implementation of multi-rigid-body dynamics within a robotic grasping simulator," in *Robotics and Automation, 2003. Proceedings. ICRA'03. IEEE International Conference on*, vol. 2, pp. 2262–2268, IEEE, 2003.

[22] E. Drumwright, "Modeling contact friction and joint friction in dynamic robotic simulation using the principle of maximum dissipation," *Algorithmic Foundations of Robotics IX*, vol. 1, no. 2000, 2011.

[23] E. Drumwright, "Moby Rigid Body Simulator." `http://physsim.sourceforge.net/`.