Proceedings of the 2nd Biennial IEEE/RAS-EMBS International
Conference on Biomedical Robotics and Biomechatronics
Scottsdale, AZ, USA, October 19-22, 2008

Design, Creation, and Validation of a Comprehensive Database Infrastructure for Robotic Grasping

Ying Li, Justin Keesling, James English, and Neil Tardella

Energid Technologies Corporation

124 Mount Auburn Street, Suite 200 North, Cambridge, MA 02138

{ying, jckeesling, jde, nmt}@energid.com

Abstract— **This paper describes the design, creation, and validation of an infrastructure for grasping. The infrastructure is premised on the ability to use large amounts of memory to store grasping algorithms in a database that can be applied broadly with any hand on any arm. This effort includes the development of a software architecture; a language syntax for configuration, shape-matching, and algorithms for grasping; and software for organizing grasp selection, configuring initial template grasps through human supervision, refining grasp placement, and refining grasp forces. An accurate dynamic simulation of a variety of robotic manipulators, hands, and objects was used to validate the approach, with positive results. The presented database approach supports generic algorithms for both fingertip and whole hand grasping of a variety of object types. Fingertip and whole hand grasps are also known as precision grasp and power grasp respectively in some literature, for example [1]. Algorithms based on grasp templates created by human supervisors and refined through automated refinement work effectively and efficiently. The presented techniques work with several types of force and touch sensors and with a variety of object shapes and physical consistencies.**

Index Terms – Robot grasping, grasp database, simulation, validation, toolkits, robotic hand.

I. INTRODUCTION

Grasping and manipulating objects is one of the most important capabilities needed for a robot to interact with the world. Its insufficiency has been identified as one of the primary obstacles to wide adoption of robots. Many techniques have been proposed for grasping, including control-based methods [2], Jacobian Techniques [3], dynamic programming [4], the use of prototypes [5], human demonstration [6], State Vector Machines, [7], shape primitives [8], and the optimization of distance metrics [9], among many others. These methods have had some specific success in the lab, but automatic generic grasping in the field is still out of reach. The best example we have for successful generic grasping is that of humans and manipulative animals. Though the functioning of mammalian brains transcends human understanding, it is clear that when presented with a new object in new context, a grasp is chosen based on stored past experience with similar objects and similar context. It is this concept that we have exploited and taken to its logical limit.

The form of the approach is a reusable software tool. Robotic-hand development companies make hardware, but typically rely on their customers to add software. These customers, therefore, are often research organizations or product companies with special and limited grasping-software development efforts. A new, generic toolkit for grasping has the potential to revolutionize both robotics and prosthetics. With this eventual goal in mind, the objectives of our research are to design and implement in C++ a comprehensive database framework for grasping rigid, soft, and articulating objects and to validate the framework through simulation using humanoid hands available today. This includes the following components:

1. The creation of a database architecture that supports the broad solution of real-world problems. The architecture must allow rapid access and fast computation for use in real time.
2. The application of XML to grasping-algorithm design. XML is a widely accepted standard language for storing and exchanging complex information. We selected and defined a specialization of XML for the grasping database through the creation of an XML schema describing a grasping language.
3. The establishment of ways to measure the quality of object matches and interpolate between near-matches to select the best grasping algorithm. This includes combining shape matching with surface-property matching to define object similarity in a new, powerful way.
4. The implementation of a caching mechanism to only load parts of the grasping database as needed in order to exploit very large (multi-terabyte) databases using only a few gigabytes of Random Access Memory (RAM).

This work was supported by NSF SBIR contract 0712300

978-1-4244-2883-0/08/$25.00 ©2008 IEEE

5. The implementation of a tool for building large grasping databases. This Database Construction Tool combines automated software with human supervision and includes human interfaces, refinement algorithms, and digital simulation.

The design of an interface for SolidWorks, a popular robot design tool that accepts common data formats for defining robotic manipulators and hands.

II. DATABASE SOFTWARE ARCHITECTURE

The grasp-algorithm database is organized into a tree structure as shown in Fig. 1, enabling the best grasping algorithm to be selected while matching the shape, articulation, and surface properties of the object to be grasped. Each leaf node in the tree provides a specific algorithm whose implementation is limited only by the interface structure and C++. Each branch node implements a fast comparison method to eliminate large portions of the tree below it. A tree structure such as this provides for virtually unlimited growth, as new algorithms for new shapes can be added without disturbing existing algorithms or adding significant unwanted computational cost.

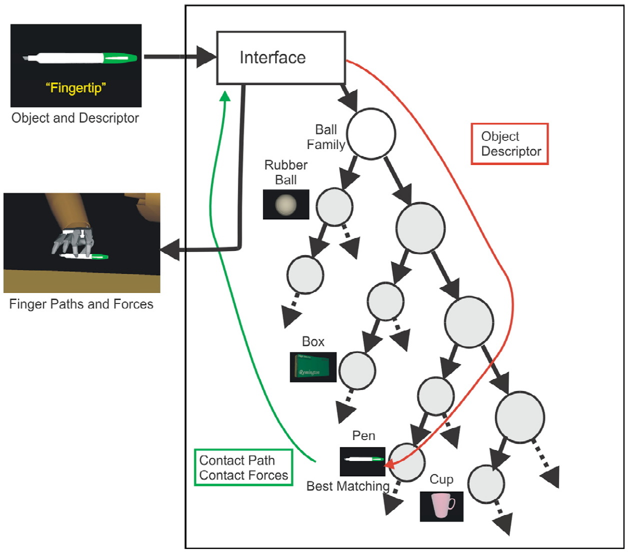


Fig. 1. The grasping approach.

The input to the database is an object description and a grasp-type descriptor. When an object is given, a sequence of increasingly narrow families is identified using the object descriptor, shape, and surface properties. The output of the database after this search is a set of finger paths and forces. An advanced new language based on XML was designed and used to represent this database.

III. DATABASE CONSTRUCTION TOOL

A significant challenge exists in creating the tool for building the grasp-algorithm database described above. The tool must be able to build a new database from a hand/manipulator description, a set of objects, and a set of environments. The organization of the tool we used is shown in Fig. 2.

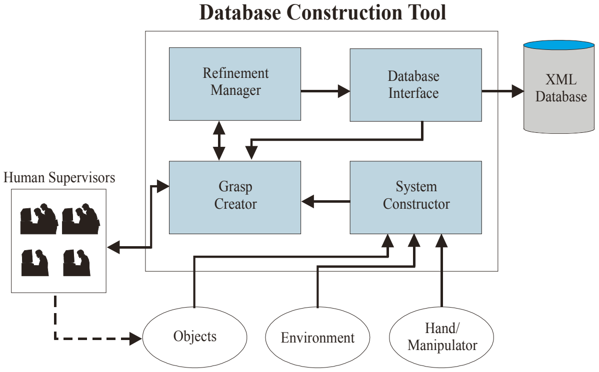


Fig. 2. Organization of the database construction tool.

At first, the system constructor will build up a world model that can represent hand/manipulator, environment instances and new object instances using some data structure. Secondly, the grasp for the object or a similar object is found in the database using a matching metric. The grasp will be presented to a supervisor if the object or a similar object exists. Here, the grasp is defined as grasping kinematic and dynamic components of the hand/manipulator. The grasp can also be generated using a human-supervised grasp creator if a similar object and its grasp cannot be found in the database. The similar grasp or generated grasp is defined as the initial grasp. Thirdly, the refinement manager refines the initial grasp to form force closure algorithmically through re-positioning better contact positions/forces with the help of human supervisors. Lastly, the refined grasp for the new object will be stored in the database. In the process of creating and refining the grasp, we take advantage of the grasp experience of human supervisors.

A. System Constructor

As shown as a component in Fig. 2, the System Constructor module builds a 3D world model that can represent the hand, manipulator, environment, and new object instances. This is implemented using Energid's software tools for robotic simulation. The System Constructor supports all robots, hands, and environmental objects, which can be kinematically redundant or bifurcating, and can have any type of joint—including rotational, prismatic, cylindrical, four-bar, and others. Objects can be grouped, and they can move freely or be attached to the environment. Of special importance, the manipulator, hand, environment, and objects to be grasped all use the same software representation and data structure. The framework supports articulated and morphing links, enabling the system to be scaled to support articulated, flexible, soft, and fragile objects. Chains, rope, pillows, and glasses, for instance, can be grasped.

B. Grasp Creator

Also shown in Fig. 2 is the Grasp Creator module. The Grasp Creator supports the creation of new grasps (in the case of completely novel objects) or refinement of existing grasps (in the case of grasps to similarly shaped objects already existing in the database). For a new object, a grasp for a similar object is searched for in the database using a matching metric. In an intuitive and repeatable procedure, the found grasp is presented to the supervisor, and the grasp is defined through both the grasping kinematic and dynamic components of the hand. Completely new grasps can be generated using a human-supervised process if a similar object and its grasp cannot be found in the database. The new grasp is then added to the database as the initial grasp for the new object. This is illustrated in Fig. 3.

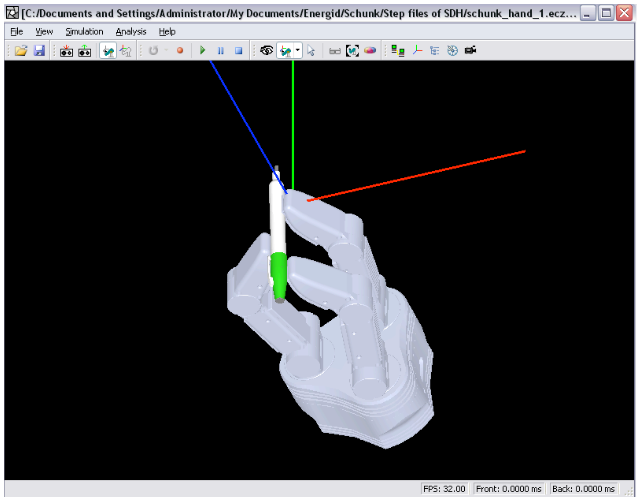
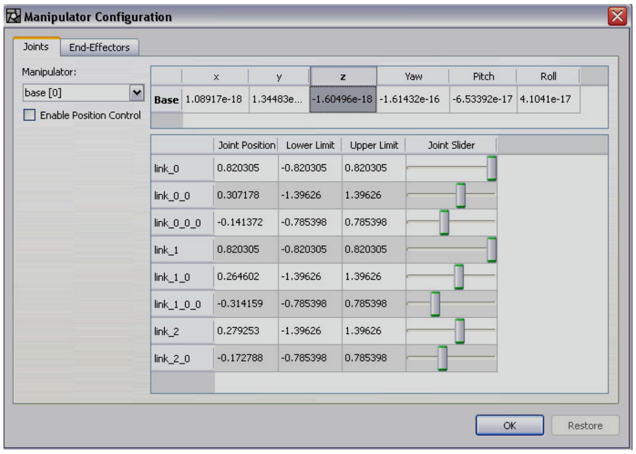


Fig. 3. An example of a supervised grasp created with our software tool. The Schunk (SDH) hand is being used to grasp a pen.

1) Software Interface

The software interface can be illustrated by example. Fig. 3 shows a human-generated grasp of a pen using a commercial Schunk hand. This grasp was generated using the sliders and intuitive configuration interface shown in Fig. 4. Joint positions and orientations were set through sliders, mouse movement, and numerical configuration. The position and orientation of the wrist can also be controlled by changing the values of x, y, z, yaw, pitch, and roll shown in the upper part of Fig. 4. The grasp can also be defined as fingertip positions in world coordinates or relative to other parts of the hand, such as the palm, as shown in Fig. 5. Hand locations, fingertip positions, and joint angles can all be controlled directly by human supervisors during grasp database construction.



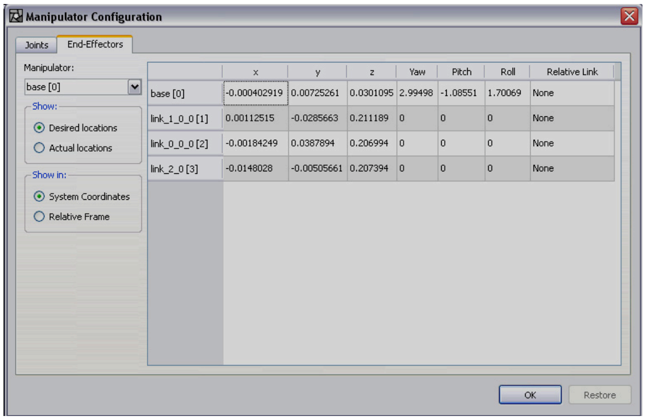
Manipulator: base [0]

☐ Enable Position Control

	x	y	z	Yaw	Pitch	Roll
Base	1.08917e-18	1.34483e...	-1.60496e-18	-1.61432e-16	-6.53392e-17	4.1041e-17

	Joint Position	Lower Limit	Upper Limit	Joint Slider
link_0	0.820305	-0.820305	0.820305	
link_0_0	0.307178	-1.39626	1.39626	
link_0_0_0	-0.141372	-0.785398	0.785398	
link_1	0.820305	-0.820305	0.820305	
link_1_0	0.264602	-1.39626	1.39626	
link_1_0_0	-0.314159	-0.785398	0.785398	
link_2	0.279253	-1.39626	1.39626	
link_2_0	-0.172788	-0.785398	0.785398	

Fig. 4. Joint-control sliders are part of the rich interface created to support human supervision of grasp construction.



Manipulator: base [0]

Show:
● Desired locations
○ Actual locations

Show in:
● System Coordinates
○ Relative Frame

	x	y	z	Yaw	Pitch	Roll	Relative Link
base [0]	-0.000402919	0.00725261	0.0301095	2.99498	-1.08551	1.70069	None
link_1_0_0 [1]	0.00112515	-0.0285663	0.211189	0	0	0	None
link_0_0_0 [2]	-0.00184249	0.0387894	0.206994	0	0	0	None
link_2_0 [3]	-0.0148028	-0.00505661	0.207394	0	0	0	None

Fig. 5. Fingertip contact positions can also be set through the software interface. Inverse kinematics algorithms are used to then set the joints.

2) Input Devices

Input devices we have found to be helpful for configuring the grasps include the P5 sensing glove, the Polhemus tracker, and the SpaceNavigator. These are shown in Fig. 6. The input framework for grasp control based on these input devices is flexible and generic. With these input devices, the human supervisor can use the best device for each stage to control the hand model in the virtual environment and to move and pre-shape the hand for creating grasps.

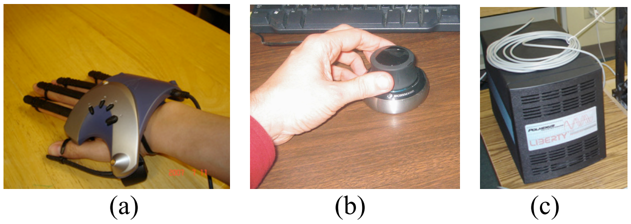


Fig. 6. Input devices that support rapid creation of the grasping database. (a) shows the P5 Virtual Hand Tracking device; (b) shows the SpaceNavigator; and (c) shows the RF-based Polhemus tracker. Energid implemented the Grasp Creator module with interfaces to these devices—the images are of the Energid systems.

3) *Grasp Alignment*

Once a hand grasp is selected from the database or generated through the supervision interface, it must be aligned to the object shape. The goal of the alignment process is to find a transformation to be applied to the hand pose so the desired contact points on the hand are brought into correspondence with points on the object [10].

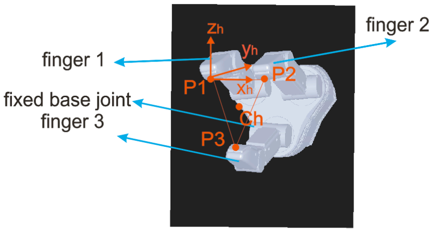
Grasp alignment algorithms of this type have been developed for use in the framework. Two simple examples are grasps for near-spherical and near-cylindrical objects using fingertips. These are illustrated in detail here for the commercial Schunk hand. The idea is to align the grasp geometry center of the hand to the geometry center of the object to be grasped.

Fig. 7 shows three contact points $\boldsymbol{P1}$, $\boldsymbol{P2}$, and $\boldsymbol{P3}$ in the palm frame as derived for the Schunk hand. A frame ($\boldsymbol{X_h}$, $\boldsymbol{Y_h}$, $\boldsymbol{Z_h}$) is generated from the three contact points. The $\boldsymbol{X_h}$ axis direction is the same as the line $\boldsymbol{P1P2}$, and the $\boldsymbol{Y_h}$ direction is pointing toward the palm from the triangle. $\boldsymbol{Z_h}$ is determined by $\boldsymbol{X_h}$ and $\boldsymbol{Y_h}$. The origin of the frame is selected as the point $\boldsymbol{P1}$. For a cylindrical grasp, the geometry center $\boldsymbol{C_h}$ is calculated as

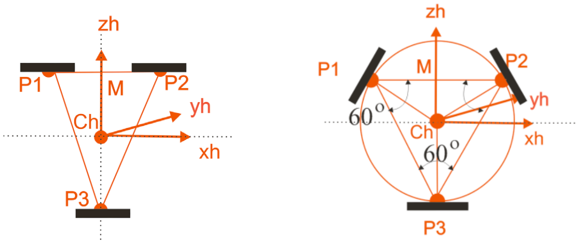
$$C_h = \frac{1}{4}P1 + \frac{1}{4}P2 + \frac{1}{2}P3 \tag{1}$$

For a sphere grasp, $\boldsymbol{C_h}$ is calculated as

$$C_h = \frac{1}{3}P1 + \frac{1}{3}P2 + \frac{1}{3}P3 \tag{2}$$



(a) A frame (X_h, Y_h, Z_h) is generated from the three contact points.



(b) Cylindrical-type grasp. **(c)** Spherical-type grasp.

Fig. 7. Pose alignment for cylindrical- and spherical-type three-finger grasps.

This tailored approach serves as a component in one of the many algorithms used to build the database shown in Fig. 1.

C. *Refinement Manager*

1) *Repositioning fingers*

After creation of an idealized grasp for a generic shape, the Refinement Manager module shown in Fig. 2 modifies the template grasp for specific object variations of surface properties and shape. It forms actual or approximate force closure algorithmically through repositioning the contacts and adjusting forces. As an example, Fig. 8 shows a refined grasp using a simulation of NASA's Robonaut [11].

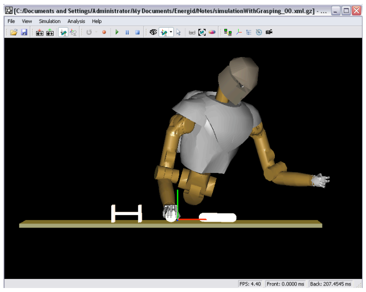
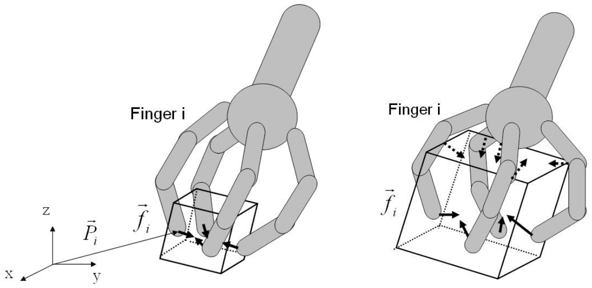


Fig. 8. Grasping refinement with Robonaut using high fidelity dynamic simulation software. First an idealized grasp is created, then this is refined using the exact object description.

2) *Force Refinement*

The Refinement Manager in Fig. 2 includes both position and force refinement. As shown in Fig. 9, a robotic hand typically has several constrained fingers with active joints which are capable of exerting force on the object to be grasped.



Fingertip Grasp Whole Hand Grasp

Fig. 9. Force refinement for two families of grasps.

The grasp modes are classified into fingertip grasps and whole hand grasps. A fingertip grasp mode is used when grasping a small object or when manipulating the object in a dexterous manner, having a small contact area at each fingertip, as shown in Fig. 9 on the left. The whole hand grasp mode, as shown in Fig. 9 on the right, is used when grasping a large object or when applying a large force to the object. Whole hand grasping gives a large contact area between the hand and the object. To create the fingertip grasp, we assume the fingers apply forces to the object through contact points. The contact points at the fingertip can exert any directional force.

After selecting nominal force values, it is necessary to modify them based on the exact object shape. A force refinement algorithm for fingertip grasping has been developed. In it, we assume the fingers apply forces to the object through the fingertip contact points shown in Fig. 10.

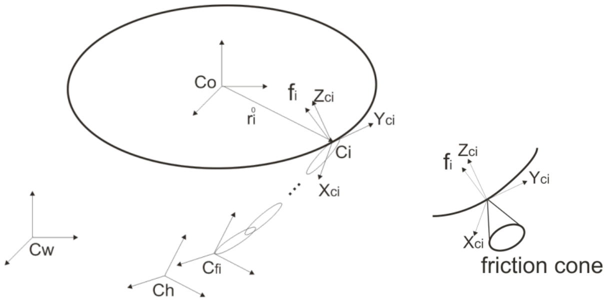


Fig. 10. Finger forces applied to the object.

Let the contact force of the hand be

$$\boldsymbol{f}_h = \begin{bmatrix} \boldsymbol{f}_1 & \boldsymbol{f}_2 & \cdots & \boldsymbol{f}_m \end{bmatrix}^T \in \mathbf{R}^{3m\times 1} \tag{3}$$

Where

$$\boldsymbol{f}_i = \begin{bmatrix} \boldsymbol{f}_{ix} & \boldsymbol{f}_{iy} & \boldsymbol{f}_{iz} \end{bmatrix}^T \tag{4}$$

The contact points at the fingertips can exert any directional forces. If the external force is defined as $\boldsymbol{F}_e$, equilibrium equations [12] for an object can be written as

$$\mathbf{G}\,\boldsymbol{f}_h + \mathbf{F}_e = \mathbf{0} \tag{5}$$

where $\mathbf{G}$ is the grasp matrix.

To achieve a stable grasp, it is expected that all the applied finger force directions are close to the contact normal of on the object. This allows an objective function for minimization to be defined as follows:

$$\mu = -\sum_{i=1}^{m} w_i \frac{f_{iz}}{f_{imax}} \tag{6}$$

where w_i is the weight for finger i. As an example, a higher weight may be given for thumb, index, and middle fingers than for pinky and ring fingers for a humanoid hand.

For force calculation, the criterion function in (6) is optimized subject to (5), friction specifications, force direction constraints, and limitations on the force angle. For a whole hand grasp, the object is enveloped by the hand. There might be many contacts between the hand and object. It is not necessary to refine the contact force over each contact to ensure individual stability. Instead, we use the interface shown in Fig. 5 refine whole-hand grasps through positioning.

Unlike in the case of fingertip grasps, whole hand grasps may have the middle links of the fingers and palm contacting the object. The forces exerted at such contact points are powerful phenomena to leverage for grasping. If no sensors are present those contact points must be regarded as passive contact points, and the forces exerted are regarded as passive contact forces. Based on this premise, we designed a force control class using only the thumb as an active contact point to apply force, while position control is applied to the other fingers and the palm. We tested applying active force with this algorithm to grasp a variety of objects. Simulation results showed that various successful grasps can be achieved with this approach. For a good grasp pose, when the active force is applied to the object through the thumb, the passive forces can be exerted at other contacts and automatically balance the active force and external force (for example, gravity) to generate a successful grasp.

D. Grasping force control system

Fig. 11 illustrates one of the force control systems used to implement the algorithms in Fig. 1. The sensor processor works with both real hardware sensors and simulated sensors. The sensor reading simulator is used to model sensor readings during simulation. The model is based on proximity measures between the manipulator and the environment. The actual force that the sensor experiences is calculated from the sensor reading and compared against the desired force for that sensor. The output of this module is the difference between the desired force and the measured force, and this value is provided to the force control module. A high bandwidth touch sensor was modeled through digital simulation. The sensor can be attached to a link, with a known location and direction with respect to the primary frame of the link. The sensor is represented by a union of convex shapes as part of the link to which it is attached. The proximity calculation routine we implemented is capable of reporting the distance query to the individual shape level.

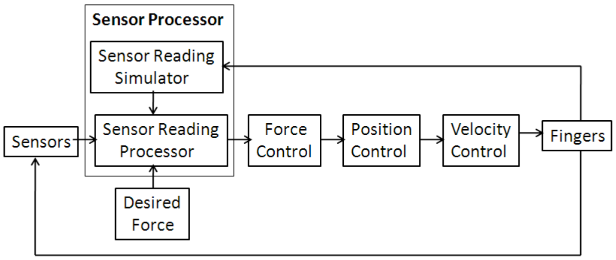


Fig. 11. A diagram of the force control system.

E. Database Interface

A key part of the database interface module shown in Fig. 2 is the shape matching algorithm. When a new object is given, using the XML-based language, the grasp for that object or a similar object is found in the database using a matching metric. This metric combines shape, articulation properties, and surface properties. Our approach is to condense the object description into a set of keys based on the most important properties of the object. The keys are defined using a variety of algorithms, including articulation analysis [13] [14], and shape analysis [15]. We analyzed a number of potential components during the study. One valuable component was a feature-based method [16] for whole object shape matching. The algorithm relies on both surface properties and the distance and angle between surface points.

IV. GENERIC ALGORITHMS FOR USE ON ANY ROBOTS

Energid Technologies' innovation is an algorithm database approach to robotic grasping tailored to real-time applications. The proposed algorithm database provides end-effector trajectories and forces to complete a grasp on both man-made and natural objects. The database is constructed using a combination of human input, object metrics, grasp algorithms, refinement algorithms, and digital simulation. It leverages the unique control algorithms provided through Energid's Actin toolkit and is described using the Extensible Markup Language (XML), with each database instantiation corresponding to one grasping mechanism, such as a pincher, hand, or pair of hands. It is one software system that supports the full spectrum of grasping mechanisms, from rudimentary grippers to complex cooperating hands. The focus is real-time operation, and in all cases, the database entries give the grasp in a fraction of a second. The database is organized to have log(*N*) access time, for *N* database entries.

V. SIMULATION RESULTS

A. Grasp Demonstrations

For the demonstrations described in this section, we used Robonaut [11], Schunk LWA [17], and Mitsubishi PA-10 [18] robotic arms and Robonaut [19] and Schunk (SDH) hands [20]. Many objects were tested for grasping. Shown in the figures are a capsule, pen, barbell, golf ball and tennis ball. Fingertip grasps are used for grasping and manipulating the tiny objects. Fig. 12 shows a PA-10 arm with a Schunk hand is used to grasp a pen. Whole hand grasps were also tested, using Robonaut to grasp a capsule, and tennis ball, as shown in Fig. 13 and Fig. 14. The grasp for these different size balls was refined algorithmically and automatically.

For the grasps in Fig. 13 and Fig. 14, the thumb was controlled in simulation to apply active force to the object. Position control was applied to the other fingers and the palm. Passive forces can be exerted at other contacts and the system can automatically balance the active forces and external forces.

Fig. 15 shows a golf ball grasp using the combination system of the Schunk arm and Robonaut hand. Only position control is applied to all the fingers and palm for the grasp, the fingertips envelop the ball from the bottom of the shape. The passive force exerted at the contact points balance the external force to grasp the golf ball successfully.

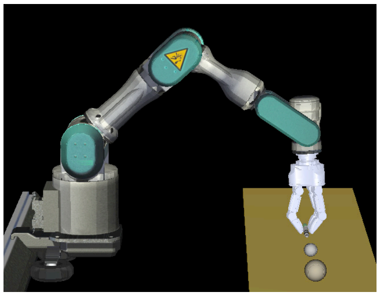


Fig. 12. PA-10 with a Schunk hand grasping a writing pen.

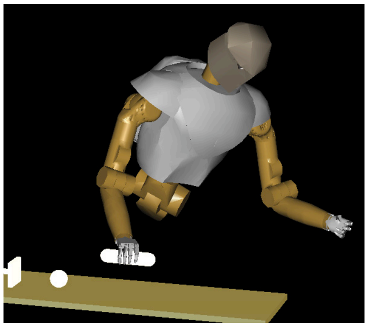


Fig. 13. Grasping a cylinder with whole hand.

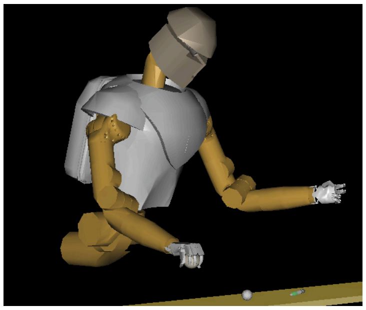


Fig. 14. Grasping a tennis ball with whole hand.

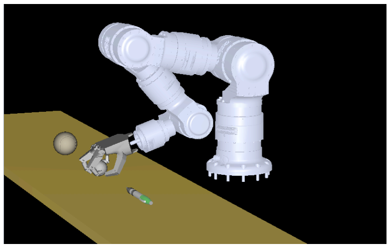


Fig. 15. Example grasp with only passive contacts using the Schunk LWA3 with the Robonaut Hand.

B. Parametric Study Tool

To quantify performance, we developed several tools, including a parametric study tool that evaluates grasp success for various configurations within a task space. Fig. 16 shows the configuration used for the study results shown in TABLE I.

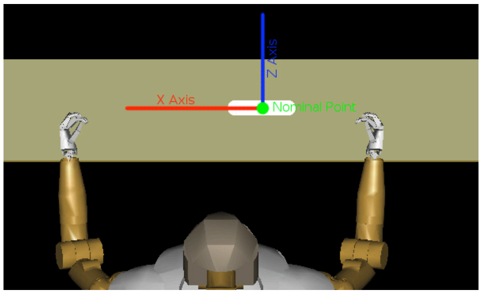
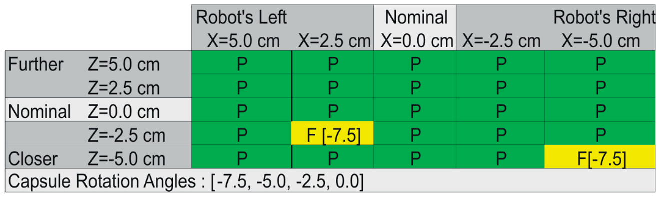


Fig. 16. Parametric study configuration.

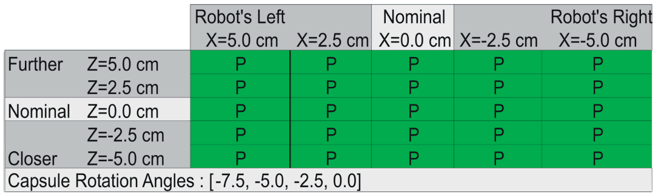
A capsule was perturbed from its nominal position in both the x and z directions (+/- 5cm, 0, +/- 2.5cm). For each of these locations, four orientations were set (0, -2.5, -5.0, -7.5 degrees), yielding a total of 100 trials. TABLE I(a) shows the results of the parametric study using an initial palm placement position. Note there are two trials where the grasp fails. The objects move away due to forces exerted with position errors. A revised grasp was defined by adding a 3mm offset to the palm in the +y (vertical) direction, and a rerun of the study gave 100% grasp success, as shown in TABLE I(b). The appropriate offset of palm from the object gives the object a small moving space during grasping, so that the hand can completely conform to the object surface and avoid errant forces. This and the other tools (such as the validation tool shown in Fig. 8) developed will play an important role in future grasp database development efforts.

TABLE I
EXAMPLE PARAMETRIC STUDY RESULTS



		Robot's Left		Nominal		Robot's Right
		X=5.0 cm	X=2.5 cm	X=0.0 cm	X=-2.5 cm	X=-5.0 cm
Further	Z=5.0 cm	P	P	P	P	P
	Z=2.5 cm	P	P	P	P	P
Nominal	Z=0.0 cm	P	P	P	P	P
	Z=-2.5 cm	P	F [-7.5]	P	P	P
Closer	Z=-5.0 cm	P	P	P	P	F[-7.5]
Capsule Rotation Angles : [-7.5, -5.0, -2.5, 0.0]						

(a)



		Robot's Left		Nominal		Robot's Right
		X=5.0 cm	X=2.5 cm	X=0.0 cm	X=-2.5 cm	X=-5.0 cm
Further	Z=5.0 cm	P	P	P	P	P
	Z=2.5 cm	P	P	P	P	P
Nominal	Z=0.0 cm	P	P	P	P	P
	Z=-2.5 cm	P	P	P	P	P
Closer	Z=-5.0 cm	P	P	P	P	P
Capsule Rotation Angles : [-7.5, -5.0, -2.5, 0.0]						

(b)

Comparison of original palm location (a) with new palm location (b). Green indicates all orientations passed, yellow indicates one failed grasp out of the three orientations, with the failed angle shown. This tool provides fast evaluation of grasp changes.

VI. CONCLUSION

In this effort we undertook the design, creation, and validation of a comprehensive infrastructure for grasping. The infrastructure supports selecting different positioning and force-control algorithms for different grasping tasks based on a tree-structured database. For populating this database, we presented several example algorithms, which represent only archetypes. Many more algorithms can be supported [21] [22]. These algorithms and the grasping infrastructure were tested within a high fidelity simulation, giving positive results.

REFERENCES

[1] Y. Yokokohji, N. Muramori, Y. Sato and T. Yoshikawa, “Designing and Encountered-type Haptic Display for Multiple Fingertip Contacts Based on the Observation of Human Grasping Behaviors,” *The International Journal of Robotics Research,* Vol. 24, No. 9, pp. 717-729, 2005.

[2] J. A. Coelho and R. A. Grupen, “A Control Basis for Learning Multifingered Grasps,” *Journal of Robotic Systems*, 14(7), pp. 545-557, 1997.

[3] R. Platt Jr., A.H. Fagg,, R.A. Grupen, “Extending Fingertip Grasping to Whole Body Grasping,” *Proceedings of the Int. Conference on Robotics and Automation (ICRA'03)*, pp. 2677-2682, 2003.

[4] N.S. Pollard, “Parallel Algorithms for Synthesis of Whole-Hand Grasps,” *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, Albuquerque, NM. 1997.

[5] N.S. Pollard, “*Parallel Methods for Synthesizing Whole-Hand Grasps from Generalized Prototypes,”* Ph.D. Thesis, Massachusetts Institute of Technology, Cambridge, Massachusetts, 1994.

[6] S.B. Kang, *Robot Instruction by Human Demonstration*, Ph.D. Thesis, Carnegie Mellon University, Pittsburgh, Pennsylvania, 1994.

[7] R. Pelossof, A. Miller, P. Allen, and T. Jebara, “An SVM Learning Approach to Robotic Grasping,” *Proceedings of the IEEE International Conference on Robotics and Automation,* New Orleans, Louisiana, pp. 3212-3218, April, 2004.

[8] A.T. Miller, S. Knoop, P.K. Allen, H.I. Christensen. "Automatic grasp planning using shape primitives." *Proceedings of the IEEE International Conference on Robotics and Automation,* Taipei, Taiwan, pp. 1824-1829, September 2003.

[9] X. Zhu and J. Wang, “Synthesis of Force-Closure Grasps on 3-D Objects Based on the *Q* Distance,” *IEEE Transactions on Robotics and Automation,* vol. 19, no. 4, August 2003.

[10] Y. Li, J. L. Fu and N. S. Pollard, “Data Driven Grasp Synthesis using Shape Matching and Task-Based Pruning,” *IEEE Transactions on Visualization and Computer graphics*, Vol. 13, pp. 732-747, 2007.

[11] R.O. Ambrose, H. Aldridge, R.S. Askew, R.R. Burridge, W. Bluethmann, M. Diftler, C. Lovchik, D. Magruder, F Rehnmark, “Robonaut: NASAapos;s space humanoid,” *Intelligent Systems and Their Applications*, Vol. 15 (4), pp.57-63, 2000.

[12] Y. Nakamura , K. Nagai, T. Yoshikawa: “Dynamics and Stability in Coordination of Multiple Robotic Mechanism,” *Int. J. Robotics Research,* vol. 8, no. 2, pp. 44-61, 1998.

[13] T. Tung and F. Schmitt, “Augmented Reeb Graphs for Content-based Retrieval of 3D Mesh Models,” *International Conference on Shape Modeling and Application*, pp157-166, 2004.

[14] D. Bespalov, W.C. Regli, and A. Shokoufandeh, “Reeb Graph Based Shape Retrieval for CAD,” *Proceedings of DETC’ 03*, Chicago, Illinois, September, 2003.

[15] J. Pu, S. Jayanti, S. Hou and K. Ramani, “Similar 3D Model Retrieval Based on Multiple Levels of Detail,” *the 14th Pacific Conference on Computer Graphics and Applications*, pp.103-112, Taipei, Taiwan, October 2006.

[16] R. Ohbuchi, T. Minamitani and T Takei, “Shape-Similarith search of 3D models by using Enhanced Shape Functions” *Theory and practice of Computer Graphics*, pp. 97-104, Birmingham, UK, June, 2003.

[17] http://robotgossip.blogspot.com/2005/12/light-weight-arm.html

[18] http://www.roboticsonline.com/buyers_guide/newproducts/details.cfm?id=301

[19] C.S. Lovchik and M. A. Diftler “The Robonaut hand: a dexterous robot hand for space,” *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 1999.

[20] http://www.schunk.com/schunk/schunk_websites/news/news_detail.html?archive=2006&article_id=7212&submenu=206&submenu2=244&country=USA&lngCode=EN&lngCode2=EN.

[21] W.-K. Chen, *Linear Networks and Systems*, CA: Wadswort T. B. Martin, R.O. Ambrose and M.A. Diftler, “Tactile Gloves for Autonomous Grasping with the NASA/DARPA Robonaut,” *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, New Orleans, LA, April 2004.

[22] K. Bernardin, K. Ogawara and K. Ikeuchi “A Sensor Fusion Approach for Recognizing Continuous Human Grasping Sequences Using Hidden Markov Models,” *IEEE TRANSACTIONS ON ROBOTICS AND AUTOMATION,* pp. 47-57, Vol.21, No. 1, 2005