2013 IEEE/RSJ International Conference on
Intelligent Robots and Systems (IROS)
November 3-7, 2013. Tokyo, Japan

A Dynamic Active Constraints Approach for Hands-On Robotic Surgery

Joshua G. Petersen and Ferdinando Rodriguez y Baena

Abstract— **Toward the goal of developing a hands-on robotic surgery control strategy which simultaneously utilizes the various strengths of both the surgeon and robot, we present a dynamic active constraint approach tailored for hands-on surgery. Forbidden region active constraints are used to prevent motion into areas which have been deemed dangerous by the surgeon, helping to overcome some of the disadvantages of fully active systems such as loss of tactile feedback, limited workspace, and limited field-of-view. The computer graphics technique of metaballs is used to represent point cloud data from an imaging system with an analytical, differentiable surface and a dynamics-based controller is proposed which controls the robot to lie on the zero set of the generated time-varying implicit function for which the motion is either known or unknown. This controller has been incorporated into a recursive null-space approach to allow for unimpeded motion along the surface and for further extension to joint optimization in the future. This methodology is demonstrated in simulation and on a lightweight, seven-degree-of-freedom serial manipulator.**

I. Introduction

Hands-on robotic surgery, in which the surgeon controls the movement of the surgical tool by directly applying forces to the robot, has been shown to offer advantages over conventional surgery and teleoperative surgery in some instances. By providing low impedance through design and control, the surgeon can feel the forces of touching or cutting directly through the robot without the need of force sensors and it has been found that controlling by directly pushing and pulling on the robot structure is intuitive for users [1]. In situations where direct force reflection and view of the operating field is vital to a successful surgery, hands-on systems are desirable over current commercial teleoperative systems which require a trade-off between the accuracy of the motion tracking and the haptic feedback provided and are limited by "the challenge of sensing forces under cost, biocompatibility and sterilizability constraints" [2].

In addition, hands-on robot systems appear to be gaining greater acceptance as the surgeon is more involved in the workflow and can control the speed and direction of the procedure [3]. It is also more clear who is in charge of the procedure, as opposed to autonomous robot systems which are off-putting for many surgeons [4].

Hands-on control can take advantage of the strengths of the surgeon and robot through the use of shared control, which is the enacting of simultaneous control policies by both the operator and robot. The overall aim of our research is to provide a shared control strategy which can combine the surgeon's medical training and understanding of the goals and decision making process necessary to complete a surgical procedure successfully and safely with the robot's accuracy, precision, and potential to have a better understanding of the physical environment through proper sensing.

To this end, we present here a dynamic active constraints control methodology for hands-on robotic surgery which can handle arbitrary and deforming constraint geometries. Using point cloud data of the constraint obtained from range data of the operating field or deformable meshes registered from CT scans, a smooth and continuous implicit surface is generated which surrounds and protects the dangerous area through the established computer graphics technique of metaballs.

We have created a dynamic controller which can direct the robot to lie on the zero set of a general implicit function and applied it to the metaball representation of the constraint surface to create dynamic active constraints. Additionally, this formulation can utilize predicted or known surface motion to more accurately track the constraint.

Lastly, we incorporate our constraint controller into a recursive null-space impedance control approach allowing us to provide unimpeded Cartesian motion and in future work, optimize the redundant degrees of freedom for obstacle avoidance and creating a more natural feedback for the surgeon.

II. Background

Forbidden region active constraints, also referred to as virtual fixtures [5], prevent or repel motion into regions which have been determined to be dangerous for the robot to enter. The use of active constraints in surgery on static geometries has been successful in both hands-on [6], [7] and teleoperative [2], [8] scenarios and has been utilized in several commercial systems (e.g. Sculptor, Stanmore Implants Worldwide Ltd. and RIO, Mako Surgical Systems Inc.)

However, many types of surgery take place in environments which are moving and deforming due to respiration, pulsation, tool contact, and other such effects. As this presents a more complicated problem in sensing and control, there has been limited research into approaches which can handle constraints in these scenarios.

Ren et al. created dynamic virtual fixtures for teleoperative beating-heart surgery using potential fields [9]. The constraints were precomputed from preoperative MR and CT images and were unable to handle unpredicted motion. One-degree-of-freedom dynamic regional constraints were created by Gibo et al. to assist in a teleoperated positioning task in

J. G. Petersen and F. M. Rodriguez y Baena are with the Mechatronics in Medicine Laboratory, Department of Mechanical Engineering, Imperial College London, London. SW7 2AZ, UK. j.petersen@imperial.ac.uk f.rodriguez@imperial.ac.uk

which the user attempted to depress an actuated soft tissue phantom to a particular depth [10]. Two methods were tested; one based on the current position of the tissue and one based on a future prediction of its position.

Rydén et al. applied their proxy-based haptic rendering of streaming point clouds to virtual fixtures of unknown geometry in teleoperation [11]. When the haptic interaction point—the desired position of the robot—penetrates the point cloud, the proxy remains a selectable distance from the surface points and forces are applied to the haptic device which are proportional to the distance between the haptic interaction point and the proxy. The method was tested in simulation and on the streaming point cloud of a beating pig heart, however the scheme was not tested using a slave robot.

To date, current approaches for dynamic active constraints have focused on teleoperation and haptic rendering, which allows for the haptic master device and slave robot to be decoupled. This simplifies the issue of forbidden region protection to position controlling the slave. Hands-on scenarios require an approach which considers the kinematics and dynamics of the robot in order to provide an appropriate impedance when constrained and a compliant response otherwise, as the surgeon is directly applying forces to the device.

III. Methodology

A. Implicit Surface Generation from Point Clouds

A variety of ways exist to define the area which the active constraints are to protect. The forbidden region can be defined as a section of a mesh, a region on a CT scan or a segmented surface identified by a 3D camera or laser scanner. The methodology presented here focuses on generating constraint surfaces from point clouds as each of the above mentioned methods can be reduced to this constraint representation.

From this point cloud, an implicit surface is produced which surrounds the area which has been chosen to be protected. The surface is smooth and continuous in order to allow for unimpeded motion of the robot along the constraint surface. In this subsection, the methodology used to generate the implicit surface is a modified version of that used by Leeper et al. [12]. In their work, implicit surfaces were used for haptic rendering of arbitrary point clouds and some of the effects of parameter choice, cloud density, and noise on the formation of the surface were analyzed. In this work, implicit surfaces are used to generate a three dimensional constraint to which the robot will be controlled.

To produce the implicit surface, compactly supported radial basis functions are used to wrap the desired constraint points as they provide several useful properties. Compactly supported radial basis functions are functions which are only non-zero within a particular region, $[0, R]$, around the point they are centered on. This allows the partitioning of large point cloud data sets in order to decrease processing time, as points which are farther than a distance R from the evaluation point will not affect the computation of the surface. Additionally, the property of compact support can be used to detect that a constraint point is approaching the surface. They also have the property $f'(0) = f'(R) = 0$ such that they provide a continuous scalar field, preventing discontinuities in the surface as a new point is approached, which is useful for providing smooth control.

The particular radial basis function used in our experiments is the Wendland function $\psi_{4,2}(r)$ [13].

$$\psi_{4,2}(r) = \begin{cases} \left(1 - \frac{r}{R}\right)^6 \left(35\left(\frac{r}{R}\right)^2 + 18\frac{r}{R} + 3\right) & r \leq R \\ 0 & r > R \end{cases} \quad (1)$$

where r is the current radius and R is the radius of influence.

An implicit function is generated using radial basis functions by finding the weighted contribution of the distance between each sensed surface point and the point which is to be constrained. In computer graphics, this is commonly referred to as a metaball surface representation [14].

$$f(x,t) = T - \sum_i \psi_{4,2}(\|x_{cp} - x_i\|) \quad (2)$$

where $f(x,t) = 0$ is the implicit surface which will be the isosurface that defines the boundary of the forbidden region, T is the threshold value, $x_{cp} \in \mathbb{R}^p$ is the current position on the robotic constraint point and $x_i \in \mathbb{R}^p$ is the i-th sensed surface point.

Methods exist which can fit an implicit surface directly to a point cloud [15], [16] however, the above method creates a surface which surrounds the constraint region. In using an offset distance, a factor of safety is included into the representation. The minimum protection distance of r_{min} can be ensured for a chosen radius of effect, R, by setting the threshold equal to,

$$T = \psi_{4,2}(r_{min}) \quad (3)$$

B. The Implicit Function Jacobian

We now have a time-deforming implicit constraint surface which is to serve as the motion constraint for the robot. From the implicit constraint function, a Jacobian is constructed, called the Implicit Function Jacobian, which is used to control the motion of the robot. First, f is required to be continuous and $\frac{df}{dt}$ and $\frac{d^2f}{dt^2}$ to exist and be continuous. This condition is satisfied by $\psi_{4,2}$ and this control methodology will work for any such constraint surface which satisfies these conditions.

Next, the time derivative of the function which defines the constraint surface is found.

$$\frac{df}{dt} = \frac{\partial f}{\partial t} + J_f \frac{dx}{dt} = 0 \quad (4)$$

where $\frac{\partial f}{\partial t} \in \mathbb{R}^p$ is the partial derivative of f explicitly with respect to time,

$$J_f = \begin{bmatrix} \frac{\partial f}{\partial x_1} & \cdots & \frac{\partial f}{\partial x_p} \end{bmatrix} \quad (5)$$

is the Jacobian of f with respect to spatial coordinates, and

$$\frac{dx}{dt} = \dot{x} = \begin{bmatrix} \dot{x}_1 \\ \vdots \\ \dot{x}_p \end{bmatrix} \tag{6}$$

is the vector of spatial variable velocities.

Since, in general, the spatial variables cannot be controlled directly, the Jacobian matrix which relates the joint velocities to the spatial variable velocities will be used.

$$\dot{x} = J\dot{q} \tag{7}$$

Substituting (7) into (4), rearranging, and normalizing yields,

$$\frac{J_f J}{\|J_f J\|}\dot{q} = -\frac{1}{\|J_f J\|}\frac{\partial f}{\partial t} \tag{8}$$

$\frac{J_f J}{\|J_f J\|}$ represents a relation between joint velocities and the velocity along the normal to the level set of the constraint function that the constraint point currently lies on. That is, if the current position of the constraint point is x_c such that $f(x_c, t) = c$ for some constant c, then $\frac{J_f J}{\|J_f J\|}\dot{q}$ will be equal to the velocity of the constraint function in the direction of the gradient at the constraint point. When the constraint point lies in the zero set of the constraint function $\{x|f(x,t) = 0\}$ and, as such, lies on the desired constraint surface, the velocities produced will be along the normal to the desired constraint surface. The Jacobian that represents this relationship, we will define to be the Implicit Function Jacobian.

$$J_{IF} \triangleq \frac{J_f J}{\|J_f J\|} \tag{9}$$

$\dot{x}_{IF}$ will also be defined to be the velocity of the constraint function in the direction of the normal to the current level set.

$$\dot{x}_{IF} \triangleq J_{IF}\dot{q} \tag{10}$$

The right hand side of (8) is the actual velocity of the surface in the direction of its normal. Perfect tracking of the time variations of the surface requires controlling the velocity of the robotic constraint point to equal this value in the direction of the normal to the constraint surface. Therefore, the desired constraint surface normal velocity, $\dot{x}_{IF}^{des}$, is defined to be equal to this value.

$$\dot{x}_{IF}^{des} \triangleq -\frac{1}{\|J_f J\|}\frac{\partial f}{\partial t} \tag{11}$$

C. Surface Constraint using the Implicit Function Jacobian

In order to control the robot to lie on the desired constraint surface, the Implicit Function Jacobian has been incorporated into the operational space approach [17] and its recursive null-space extension [18]. The operational space approach transforms the joint dynamics into the dynamics of task space in order to create controllers which are linear in the accelerations of the chosen task.

The dynamics of a robot with n joints can be expressed in the Lagrangian formalism as

$$A(q)\ddot{q} + b(q, \dot{q}) + g(q) = \Gamma \tag{12}$$

where A is the inertia matrix, b is the Coriolis-centrifugal vector, g is the gravity vector, and Γ is the generalized force vector.

For some arbitrary task $x_t = T(q)$ and its instantaneous kinematics $\dot{x}_t = J_t\dot{q}$ (where J_t is the task Jacobian), the task space dynamics can be computed by

$$\overline{J}_t^T(A\ddot{q} + b + g = \Gamma) \Rightarrow \Lambda_t\ddot{x}_t + \mu_t + p_t = F_t \tag{13}$$

where $\overline{J}_t = A^{-1}J_t^T\Lambda_t$ is the dynamically consistent inverse of J_t, $\Lambda_t = (J_t A^{-1} J_t^T)^{-1}$ is the operational space inertia matrix, $\mu_t = \overline{J}_t^T b - \Lambda_t\dot{J}_t\dot{q}$ is the operational space Coriolis-centrifugal vector, $p_t = \overline{J}_t^T g$ is the operational space gravity vector, and F_t is the vector of operational space forces.

By appropriately choosing $F_t = \Lambda_t\ddot{x}_t^{ref} + \mu_t + p_t$, where $\ddot{x}_t^{ref}$ is the reference acceleration control input, the dynamics of the robot can be linearized in the task space. These operational space forces are mapped back to actuation torques through the transpose of the Jacobian.

$$\Gamma = J_t^T F_t \tag{14}$$

Using the operational space technique we can transform the dynamics into the space of the constraint using J_{IF}. Now suitable low level acceleration controllers must be defined, which will control the constraint point to lie on the desired surface. Two cases arise when constraining the robot; constraint surfaces whose motion is known or predicted and constraint surfaces which are static or whose motion is unknown.

The motion of known time-varying surfaces can be tracked by using a feed-forward controller which compensates for the normal velocity and acceleration of the surface.

$$\ddot{x}_{IF}^{ref} = \ddot{x}_{IF}^{des} - K_p f(x_c, t_c) + K_d(\dot{x}_{IF}^{des} - J_{IF}\dot{q}) \tag{15}$$

where

$$\ddot{x}_{IF}^{des} \triangleq -\frac{d}{dt}\left(-\frac{1}{\|J_f J\|}\frac{\partial f}{\partial t}\right) \tag{16}$$

is the acceleration of the surface in the direction of its normal, x_c is the current position, t_c is the current time, K_p is a diagonal matrix of position gains, and K_d is a diagonal matrix of velocity gains.

If time deformations are unknown, we can control to the current estimate of the surface using the following controller.

$$\ddot{x}_{IF}^{ref} = -K_p f(x_c) - K_d J_{IF}\dot{q} \tag{17}$$

In either case, the controller which will control the robot along the level sets of the implicit constraint to the zero set is,

$$\Gamma = J_{IF}^T F_{IF} = J_{IF}^T(\Lambda_{IF}\ddot{x}_{IF}^{ref} + \mu_{IF} + p_{IF}) \quad (18)$$

In addition to controlling the robot to lie on the constraint surface, the hands-on controller should provide unimpeded motion along the surface and provide a posture for the remaining degrees of freedom. This is accomplished with the recursive extension to the operational space approach [18], which allows for a hierarchy of an arbitrary number of tasks to simultaneously operate. Each task in the hierarchy operates in the degrees of freedom left out from higher order tasks by operating in the null space of the previous Jacobians. This controller takes the form,

$$\Gamma = \sum_{k=1}^{N} J_{k|prec(k)}^T F_{k|prec(k)} \quad (19)$$

where $J_{k|prec(k)} = J_k N_{prec(k)}$ is the Jacobian of the k-th task operating in the null space of the previous $k-1$ tasks, $N_{prec(k)} = I - \sum_{k=1}^{N} \overline{J}_{k|prec(k)} J_{k|prec(k)}$ is the combined null space of the higher order tasks, and $F_{k|prec(k)}$ is the forces of the k-th task acting in the null space of the previous $k-1$ tasks.

The three-task prioritized controller which hierarchically constrains the robot, allows motion along the surface of the constraint, and controls the residual degrees of freedom is

$$\Gamma = J_{IF}^T F_{IF} + J_{pt|IF}^T F_{pt|IF} + J_{p|pt|IF}^T F_{p|pt|IF} \quad (20)$$

where J_{pt} is a two degree of freedom Cartesian velocity Jacobian and J_p is the Jacobian of the posture. $F_{pt|IF}$ controls the forces in the Cartesian directions along the surface and $F_{p|pt|IF}$ controls the forces in the residual posture.

D. Experimental Setup

The experimental setup used is shown in Figure 1. A Kuka LWR 4+ was controlled through the Fast Research Interface (FRI, Kuka Robotics GmbH) with a PC using Ubuntu 10.04, OROCOS (orocos.org) and ROS Electric (ros.org). Torques were commanded in the FRI operating in the axis-specific impedance control mode with the virtual spring stiffness and damping parameters set to zero. We modified the open source Whole-Body Control Software to create our controllers [19]. A dynamic soft tissue brain phantom made of Platsil Gel was created and actuated using a linear actuator controlled by an Arduino Uno. An Optotrak Certus (Northern Digital Inc.) was used to track two markers attached to the outer surface of the brain phantom and the generated point clouds were sent to the Controller via the User Datagram Packet over Internet Protocol (UDP/IP). The surface variations were approximately 1cm in amplitude and had a frequency of about 0.45 Hz.

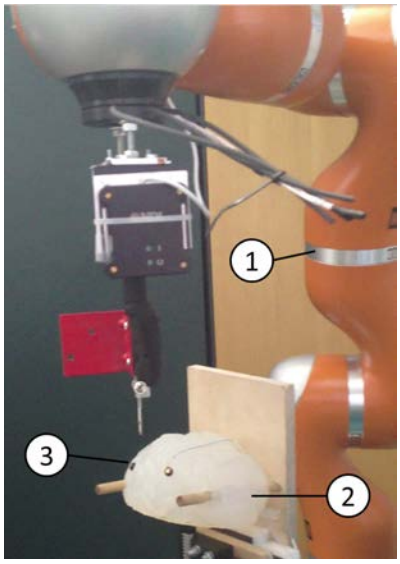

Fig. 1. Kuka LWR 4+ (1) with dynamic soft tissue phantom (2) and attached Optotrak markers (3)

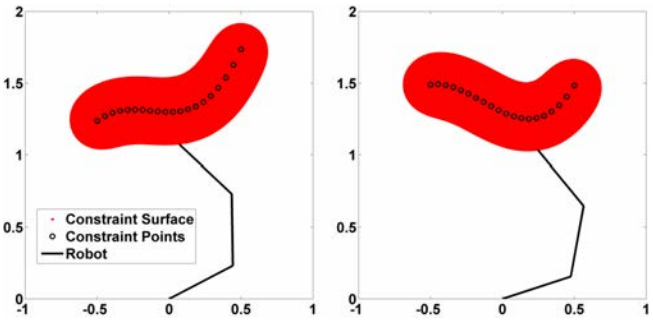

Fig. 2. Robot and unpredicted constraint surface at t = 2.5s and t = 12.5s

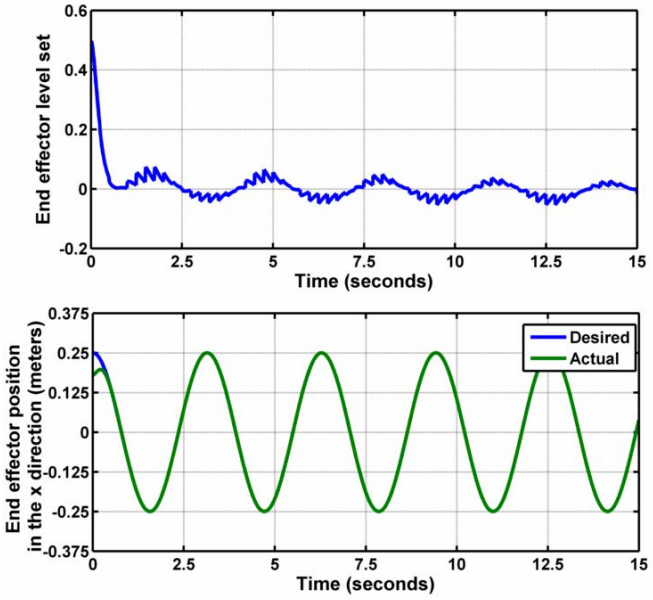

Fig. 3. Unpredicted surface results depicting the level set of the end effector (a) and the x position of the end effector (b)

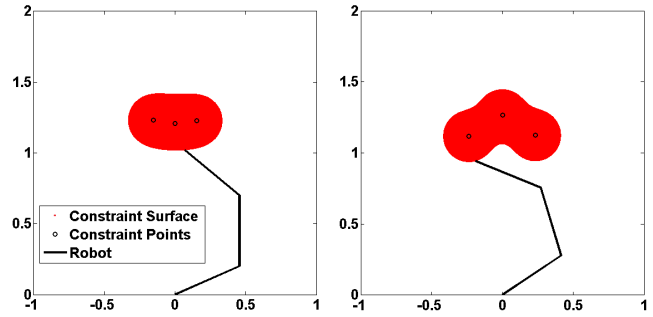

Fig. 4. Robot and predicted constraint surface at t = 2.5s and t = 7.5s

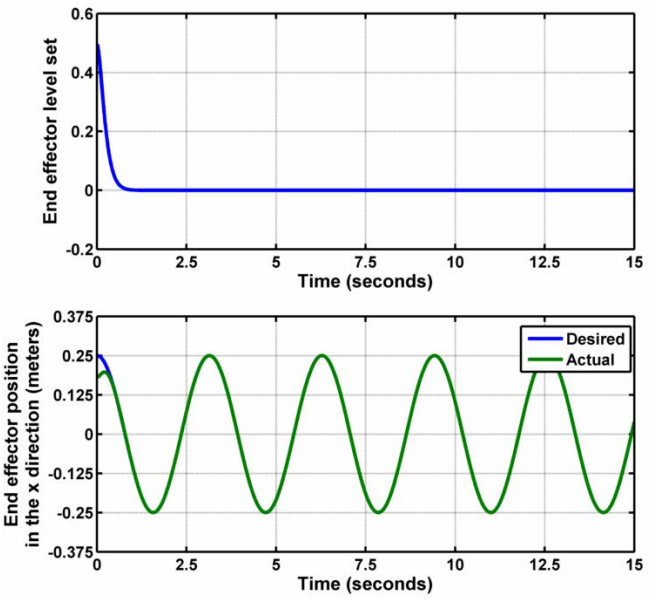

Fig. 5. Predicted surface results depicting the level set of the end effector (a) and the x position of the end effector (b)

IV. Experiments and Results

A. Simulation

A 2D dynamic simulation was implemented in Matlab R2012a (Mathworks Inc.) to test our control algorithm. The controller was first tested on a surface updated at 40 Hz for which the deformations were unknown a priori. In addition to constraining the end effector to the surface, a feed forward controller was used to control the x coordinate of the position of the end effector to follow a sinusoid. Lastly, we commanded a default posture of the zero position.

Figure 2 depicts the simulated robot kinematic chain, constraint points and constraint surface at two time intervals during the simulation. In Figure 3a, the error in the level set of the constraint is shown. Due to the discrete nature of the range data, at every update of the camera the error jumps to a non zero value and the controller performs gradient descent along the level sets of the implicit constraint function. This does not affect the positioning of the end effector in the x direction, as shown in Figure 3b, which is still able to track the desired trajectory.

Additionally, a 2D simulation was created for testing the controller operating on a constraint surface in which the motion of the points was known. Each constraint point's x and y position were defined to be known sinusoids. Similarly to the previous test, we constrain the motion to the constraint surface, a sinusoid trajectory is commanded for the x position of the end effector, and the posture is command to the zero position.

Figure 4 shows the robot and constraint at two different times during the simulation. Figure 5a shows that the controller performs an approximately critically damped response to a level set of zero while Figure 5b shows that the controller simultaneously tracks the desired position trajectory.

B. Kuka LWR Implementation

A simple two-state controller for hands-on use was created for the Kuka LWR 4+. When away from a constraint surface, the following controller which consisted of a Cartesian position controller and a joint controller in the null space was used.

$$\Gamma = J_{pt}^T F_{pt} + J_{p|pt}^T F_{p|pt} \tag{21}$$

where J_{pt} is the Cartesian velocity Jacobian, F_{pt} is the Cartesian force, $J_{p|pt}$ is the joint or postural Jacobian acting in the null space of the Cartesian Jacobian, and $F_{p|pt}$ is the force of the posture acting in the null space of the Cartesian velocity Jacobian.

When the user approached the surface with the end effector and reached a particular level set threshold, the controller switched to the surface constraint controller (20). Similarly, the user could exert a force to move the tool away from the surface and, when the tool reached another level set threshold, the controller switched back to (21).

In both controllers, we set the Cartesian gains to zero to allow for the user to freely control the tool in Cartesian space and along the surface. The acceleration level postural controllers were set to damp out any motion in the posture, allowing the user to maneuver the postural degrees of freedom if necessary, while ensuring they approximately held their position while the user was controlling the end effector. Additionally, as opposed to the simulations, the Coriolis-centrifugal forces were not compensated for on the LWR. However, due to the low velocity nature of these experiments, this did not affect the results significantly.

Figure 6 shows the error to the constraint surface during the experiment utilizing the phantom and Figure 7 shows the magnitude of force the user applied to the robot during the test. At the start, the tool was away from the surface and no torques were commanded to constrain the tool. The user moved the tool towards the moving constraint surface and at time t = 3.85s, the tool passed the level set limit, engaging the constraint. At approximately 25.5 seconds, the user exerted a force on the tool away from the constraint and at t = 26.15s the constraint disengaged, allowing unimpeded movement of the end effector.

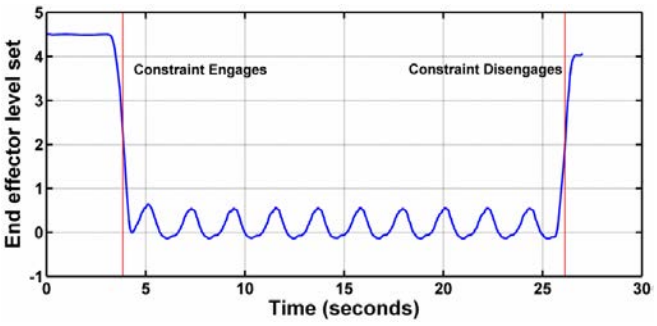

Fig. 6. Dynamic phantom results depicting level set of the end effector

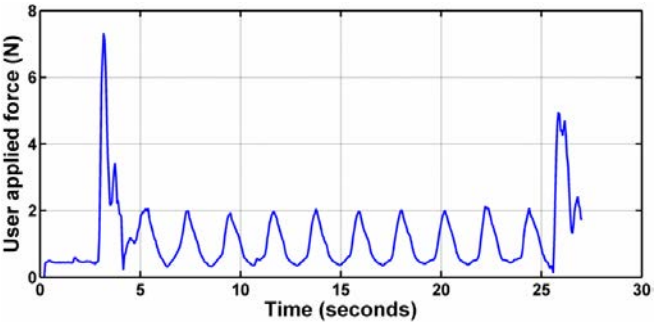

Fig. 7. Magnitude of user applied forces during dynamic phantom tests

V. Conclusions and Future Work

We have presented a representation of dynamics active constraints intended for hands-on robotic surgery. This methodology used the computer graphics technique of metaballs to produce a smooth and continuous implicit function which surrounded the points describing the surface to be constrained, be they vertices of a deforming 3D mesh or a point cluster acquired via intraoperative range imaging. In order to control the robot to lie on the zero set of this constraint function, a dynamic controller was derived that could also include predicted or known surface motion to track the moving constraint more accurately. Lastly, this control strategy was incorporated into a recursive null-space approach to allow for free motion along the surface of the constraint and for future optimization of the residual degrees of freedom. The feasibility of the approach was demonstrated in simulation and on an integrated robotic system.

Future work on this active constraint approach will focus on testing the feasibility of the controller which utilizes prediction of the surface motion on a robotic system, investigating the effects of the user on the control hierarchy, and considering methods for constraining the orientation of the end effector. Additionally, towards the goal of creating an overall shared control approach for hands-on robotic surgery, null-space postural optimizations will be investigated for obstacle avoidance and for more natural surgical motion.

Acknowledgments

James Southall-Andrews created the dynamic soft-tissue phantom and implemented the code for the Optotrak Certus. Experiments were conducted at the Neuroengineering and Medical Robotics Laboratory at Politecnico Di Milano. This work was supported by EU Grant FP7-ICT-2009-6-270460.

References

[1] T. Ortmaier, H. Weiss, U. Hagn, M. Grebenstein, M. Nickl, A. Albu-Schaffer, C. Ott, S. Jorg, R. Konietschke, L. Le-Tien, and G. Hirzinger, "A hands-on-robot for accurate placement of pedicle screws," in *Robotics and Automation, 2006. ICRA 2006. Proceedings 2006 IEEE International Conference on*, May 2006, pp. 4179 –4186.

[2] T. Yamamoto, N. Abolhassani, S. Jung, A. M. Okamura, and T. N. Judkins, "Augmented reality and haptic interfaces for robot-assisted surgery," *The International Journal of Medical Robotics and Computer Assisted Surgery*, vol. 8, no. 1, pp. 45–56, 2012.

[3] M. Jakopec, S. Harris, F. Rodriguez y Baena, P. Gomes, J. Cobb, and B. Davies, "Preliminary results of an early clinical experience with the acrobot system for total knee replacement surgery," in *Medical Image Computing and Computer-Assisted Intervention MICCAI 2002*, 2002, vol. 2488, pp. 256–263.

[4] B. Davies, S. Harris, F. Rodriguez y Baena, P. Gomes, and M. Jakopec, "Hands-on robotic surgery: Is this the future?" in *Medical Imaging and Augmented Reality*, 2004, vol. 3150, pp. 27–37.

[5] L. B. Rosenberg, "The use of virtual fixtures as perceptual overlays to enhance operator performance in remote environments," USAF Armstrong Laboratory, Tech. Rep., 1992.

[6] S. Ho, R. Hibberd, and B. Davies, "Robot assisted knee surgery," *Engineering in Medicine and Biology Magazine, IEEE*, vol. 14, no. 3, pp. 292 –300, May/June 1995.

[7] A. Bettini, P. Marayong, S. Lang, A. M. Okamura, and G. D. Hager, "Vision-assisted control for manipulation using virtual fixtures," *IEEE Trans. Robot.*, vol. 20, no. 6, pp. 953–966, Dec 2004.

[8] N. Turro, O. Khatib, and E. Coste-Maniere, "Haptically augmented teleoperation," in *Proc. of the IEEE International Conference on Robotics and Automation*, vol. 1, Seoul, South Korea, May 2001, pp. 386–392.

[9] J. Ren, R. Patel, K. McIsaac, G. Guiraudon, and T. Peters, "Dynamic 3-d virtual fixtures for minimally invasive beating heart procedures," *Medical Imaging, IEEE Transactions on*, vol. 27, no. 8, pp. 1061 – 1070, Aug 2008.

[10] T. Gibo, L. Verner, D. Yuh, and A. Okamura, "Design considerations and human-machine performance of moving virtual fixtures," in *Robotics and Automation, 2009. ICRA '09. IEEE International Conference on*, May 2009, pp. 671 –676.

[11] F. Ryden and H. Chizeck, "Forbidden-region virtual fixtures from streaming point clouds: Remotely touching and protecting a beating heart," in *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, Oct 2012, pp. 3308 –3313.

[12] A. Leeper, S. Chan, and K. Salisbury, "Point clouds can be represented as implicit surfaces for constraint-based haptic rendering," in *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, May 2012, pp. 5000 –5005.

[13] H. Wendland, "Piecewise polynomial, positive definite and compactly supported radial functions of minimal degree," *Advances in Computational Mathematics*, vol. 4, no. 1, pp. 389–396, December 1995.

[14] J. F. Blinn, "A generalization of algebraic surface drawing," *ACM Trans. Graph.*, vol. 1, no. 3, pp. 235–256, July 1982.

[15] V. V. Savchenko, A. A. Pasko, O. G. Okunev, and T. L. Kunii, "Function representation of solids reconstructed from scattered surface points and contours," *Computer Graphics Forum*, vol. 14, no. 4, pp. 181–188, 1995.

[16] B. S. Morse, T. S. Yoo, D. T. Chen, P. Rheingans, and K. R. Subramanian, "Interpolating implicit surfaces from scattered surface data using compactly supported radial basis functions," in *Proceedings of the International Conference on Shape Modeling & Applications*, ser. SMI '01. Washington, DC, USA: IEEE Computer Society, 2001.

[17] O. Khatib, "A unified approach for motion and force control of robot manipulators: The operational space formulation," *IEEE Journal of Robotics and Automation*, vol. RA-3, no. 1, pp. 43–53, February 1987.

[18] L. Sentis, "Synthesis and control of whole-body behaviors in humanoid systems," Ph.D. dissertation, Stanford University, July 2007.

[19] R. Philippsen, L. Sentis, and O. Khatib, "An open source extensible software package to create whole-body compliant skills in personal mobile manipulators," in *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, San Francisco, CA, USA, September 2011, pp. 1036–1041.