2017 IEEE International Conference on Systems, Man, and Cybernetics (SMC)
Banff Center, Banff, Canada, October 5-8, 2017

Laparoscopic Instrument Tip Position Estimation for Visual and Haptic Guidance in the Computer Assisted Surgical Trainer

Minsik Hong[1], Kuo-Shiuan Peng[1], Victoria Lundine[2], and Jerzy W. Rozenblit[1, 3]
[1]Department of Electrical and Computer Engineering, [2]Biomedical Engineering, and [3]Department of Surgery
University of Arizona, Tucson, Arizona, USA
Email: {mshong, kspeng, victorialundine}@email.arizona.edu, and jr@ece.arizona.edu

***Abstract*— A mathematical model of 4 degrees of freedom of forward and inverse kinematics is presented to provide visual and haptic guidance for laparoscopic surgery skills training. Also, a particle swarm optimization technique with color marker detection is used to estimate the instrument tip position accurately. Using a simple heuristic for the visual guidance, inverse kinematics solutions are calculated in real-time, which the haptic guidance uses as well. The experimental results illustrate the effectiveness of the proposed method.**

Keywords—Surgical Training; Visual Guidance; Haptic guidance; Inverse Kinematics; Particle Swarm Optimization

I. Introduction

Minimally invasive surgery generally provides several benefits such as fast recovery time, fewer incisions, and minimial blood loss to patients. However, becoming an expert surgeon is challenging due to mainly using special surgical instruments such as endoscope camera and graspers. The challenges are restricted vision, hand-eye coordination problem, and lack of tactile sensation. Therefore, the trainee should practice using training devices and earn a certificate (e.g., the Fundamentals of Laparoscopic Surgery (FLS) designed by Society of American Gastrointestinal and Endoscopic Surgeons (SAGES) in the United States). The FLS program, designed for basic laparoscopic surgery skills training, provides five hands-on exams [1].

The FLS trainer box was designed to practice such exam tasks [2]. This trainer box is relatively inexpensive and has simple training kits (e.g., there is no active learning support such as visual and force guidance). In contrast, several virtual reality (VR)-based training simulators are proposed to provide a better training environment by providing an assessment report or providing tactile force feedback [3]-[6].

We are developing innovative techniques for a computer-guided laparoscopic surgical training system called Computer Assisted Surgical Trainer (CAST) [7]. The short term vision is to transfer the technology to training centers for better surgical outputs and improved patient safety. Unlike existing VR training simulators, CAST physically guides trainees' instruments during a particular training session through assistive force with an augmented reality display. It consists of two mechanical fixtures, a scene box with a web camera that provides a realistic training environment, electronics such as motors and encoders, and four software modules. Fig. 1 illustrates the overall system. To guide a trainee, a collision free and shortest recommended path is generated for a surgical instrument's movement [8]. CAST guides a trainee by providing both visual [9] and haptic guidance (e.g., applying force to minimize the deviation from a desired path using forbidden region virtual fixtures) [10] based on the generated path. While performing a particular task, movement data is collected to evaluate training performance [11].

In this paper, we first derive a forward kinematics model and present how to find the best parameters for the forward kinematics model using Particle Swarm Optimization (PSO) with image-based color object tracking. The derived model is used for rendering guidance cues. For the haptic guidance, inverse kinematics solutions are needed to design motor controllers. We propose an application specific algorithm to calculate the inverse kinematics solutions in real-time. The proposed algorithm is used to generate a reference for both visual and haptic guidance.

The rest of this paper is organized as follows: a mathematical model of forward and inverse kinematics with a parameter optimization method is presented in Section II. In Section III, a reference generation method for visual and haptic

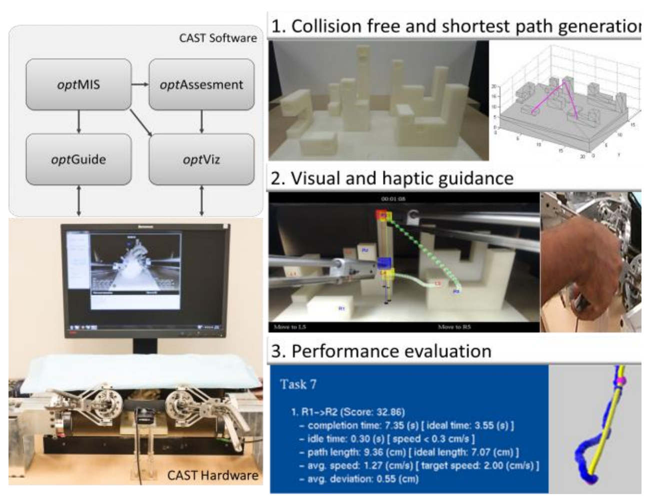


Figure 1. CAST overview.

Figure 2. A mechanical fixture in CAST.

guidance is explained. Experimental results are shown in Section IV. Finally, the conclusion is given in Section V.

II. KINEMTACIC CONFIGURATION

The CAST mechanical platform (whose prototype was produced for us by Manufacturing Systems Solutions, LLC) consists of two aluminum fixtures. Each fixture has a gimbal that allows four degrees of freedom (i.e., yaw, pitch, roll, and insertion). A standard laparoscopic instrument is mounted on the fixture. In laparoscopic surgery, a trocar is used to insert various surgical instruments into a human body while performing surgery. The gimbal not only enables the trocar's functionality to be imitated, but also enables expression of real instrument movements. Fig. 2 depicts the right fixture. Stainless wires are used to connect the gimbal joints and the corresponding motor and encoder. The motors are used for force guidance.

A. Forward kinematics

The instrument tip position is expressed using Euler angles with an insertion length. Three Euler angles, θ, ϕ, and ψ, describe yaw, pitch, and roll, respectively. The insertion length (L) represents how deep the instrument moves from the gimbal center. In [7], the roll was ignored because we assume that the instrument is on the roll axis to make the model simple. By considering yaw, pitch, and insertion movement, the estimated tip position ($\hat{P}_{tip}$) is expressed as follows:

$$\hat{P}_{tip} = R_X(\theta)R_Z(\phi)\begin{bmatrix} L \\ 0 \\ 0 \end{bmatrix} + P_G \qquad (1)$$

where $\hat{P}_{tip}$ is a 3×1 vector, $R_X(\theta)$ is a 3×3 rotation matrix that rotates a 3×1 vector by θ around the x-axis, $R_Z(\phi)$ is a 3×3 rotation matrix that rotates a 3×1 vector by ϕ around the z-axis, and P_G is a 3×1 vector that represents the gimbal

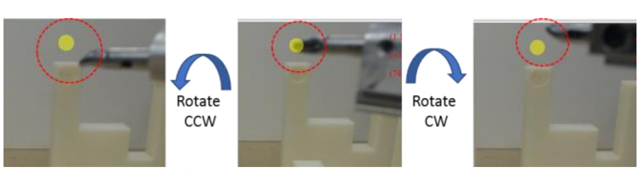


Figure 3. Example of position estimation where yellow circle represents calculated position using (1).

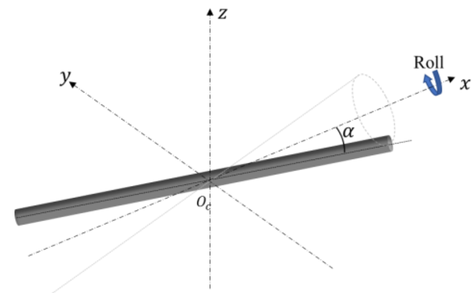


Figure 4. A modified model that takes tilting into account.

center position.

However, unlike our previous assumption, the instrument is slightly tilted from the rotation axis due to manufacturing errors. As a result, the roll angle significantly affects the position estimation results as shown in Fig. 3. Also, it is necessary to consider roll motion for more complicated training scenarios. Therefore, modification of (1) is required in order to take into account the roll motion with a tilting angle. Fig. 4 illustrates a modified model. The tilting angle (α) is used to model the tilting effect (i.e., the instrument is rotated by α around the z-axis). By considering roll angle (ψ) and tilting angle (α), (1) is replaced as follows:

$$\hat{P}_{tip} = R_X(\theta)R_Z(\phi)R_X(\psi)R_Z(\alpha)\begin{bmatrix} L \\ 0 \\ 0 \end{bmatrix} + P_G \qquad (2)$$

where α is a constant value. The three Euler angles (θ, ϕ, and ψ) and the insertion length (L) are calculated using the encoder counter values with several parameters as shown below.

$$\Omega = \varepsilon \cdot \Delta\Omega / \Delta\varepsilon + \Omega_{\min}$$

where $\Omega \in \{\theta, \phi, \psi, L\}$, Ω is restricted to $[\Omega_{\min}, \Omega_{\max}]$, $\Delta\Omega = \Omega_{\max} - \Omega_{\min}$, ε is the corresponding encoder counter value and its range is $[\varepsilon_{\min}, \varepsilon_{\max}]$, and $\Delta\varepsilon = \varepsilon_{\max} - \varepsilon_{\min}$. For instance, the yaw angle is calculated as follows:

$$\theta = \varepsilon_{yaw} \cdot \Delta\theta / \Delta\varepsilon_{yaw} + \theta_{\min}$$

where $\Delta\theta = \theta_{\max} - \theta_{\min}$, ε_{yaw} is the present encoder counter value for yaw-axis, and $\Delta\varepsilon_{yaw} = \varepsilon_{yaw,\max} - \varepsilon_{yaw,\min}$.

To estimate the tip position accurately, the optimal parameters need to be found. For each Ω, there are four parameters (i.e., $\Omega_{\min}, \Omega_{\max}, \varepsilon_{\min}$, and $\varepsilon_{\max}$) that are used to calculate the corresponding angle or length. Given four encoder counter values (i.e., $\varepsilon_{yaw}, \varepsilon_{ins}, \varepsilon_{pit}$, and ε_{rol}) with 16 parameters, (2) is redefined as follows:

$$\hat{P}_{tip} = (\hat{x}, \hat{y}, \hat{z}) = g(\varepsilon_{yaw}, \varepsilon_{ins}, \varepsilon_{pit}, \varepsilon_{rol} | \Theta) \qquad (3)$$

where Θ is a set that contains 16 parameters, three gimbal center position values (i.e., (x_G, y_G, z_G)), and tilting angle (α). The total number of parameters is 20. If the best parameters are found, they will be guaranteed to accurately estimate the tip position.

B. Parameter optimization

To optimize the parameters (Θ) of the function g, Particle Swarm Optimization (PSO) is used. As a bio-inspired optimization technique, PSO has been widely used because it is easy to use, requires less tuning parameters and less computation, and is good to use in high dimensional space problems [12]. The details of PSO algorithm with various parameter selection methods were presented in [12].

The fitness function to optimize Θ is defined as follows:

$$f = \frac{1}{m}\sum_{i=1}^{m}\left\| g\left(\varepsilon_{yaw}^{i}, \varepsilon_{ins}^{i}, \varepsilon_{pit}^{i}, \varepsilon_{rol}^{i} \middle| \Theta\right) - P_{tip}^{i}\right\| \tag{4}$$

where i is the index to represent the i^{th} sample, m is the number of samples, and P_{tip}^{i} is the i^{th} actual tip position given corresponding four encoder counter values. The objective of the optimization is to find Θ to minimize f (i.e., minimize the average of the distance errors among the estimated positions and the actual positions).

We can use the fitness function f with data pairs (i.e., actual positions with encoder counter values) that are collected by asking a user to touch pre-defined targets. However, it is challenging for a user to point to the exact pre-defined targets without error. To provide a better user interface, color marker detection is implemented. A primary color marker (e.g., red marker) is attached on the instrument tip to trace the instrument tip as shown in Fig. 5-(a). The original color image is converted to YCbCr image [13]. To extract the primary color marker area, a threshold in YCbCr domain is used. The tip position in the image plane is estimated by calculating the center of the extracted marker area as shown in Fig 5-(b).

Using a camera matrix [14], we can map a position in the real world on the image plane. This camera matrix should be used for the visual guidance in CAST. The mapping function is defined as follows:

$$Q = \begin{bmatrix} x_p \\ y_p \end{bmatrix} = h(P|\Gamma) = \begin{bmatrix} u/w \\ v/w \end{bmatrix}, \begin{bmatrix} u \\ v \\ w \end{bmatrix} = M\begin{bmatrix} P \\ 1 \end{bmatrix} \tag{5}$$

where P is a 3×1 vector representing a real world position, Q is a 2×1 vector representing the corresponding position in the image plane, M is a 3×4 camera matrix, and Γ is the elements of the camera matrix. Given encoder counter values, the estimated position ($\hat{x}_p$, $\hat{y}_p$) in the image plane is calculated by using (3) and (5).

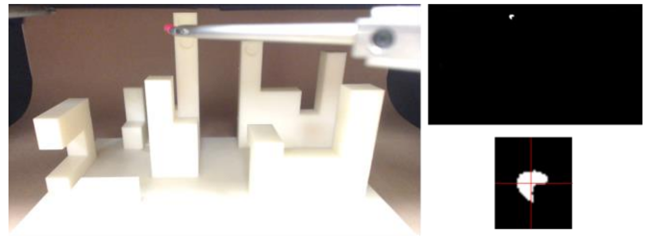


Figure 5. (a) An instrument with red color marker, (b) detection result.

$$\begin{bmatrix} \hat{x}_p \\ \hat{y}_p \end{bmatrix} = h\left(\hat{P}_{tip}|\Gamma\right) = h\left(g\left(\varepsilon_{yaw}, \varepsilon_{ins}, \varepsilon_{pit}, \varepsilon_{rol}|\Theta\right)|\Gamma\right) \tag{6}$$

Using (6), the fitness function for PSO is redefined as follows:

$$\tilde{f} = \frac{1}{m}\sum_{i=1}^{m}\left\| h\left(g\left(\varepsilon_{yaw}^{i}, \varepsilon_{ins}^{i}, \varepsilon_{pit}^{i}, \varepsilon_{rol}^{i}|\Theta\right)|\Gamma\right) - Q_{tip}^{i}\right\|$$

where Q_{tip}^{i} is the i^{th} actual tip position on image plane, which is determined by color marker detection.

The brief measurement data are used to generate initial population. Also, human knowledge is used to limit the search space (e.g., a user can measure the initial insertion length $L_{\min}$ with a measurement error. This $L_{\min}$ should be bounded to [$L_{\min} - e$, $L_{\min} + e$] where e is a reasonable error value like 1cm.). In [15], several bound handling methods were discussed for PSO in high-dimensional search spaces. Random scheme [16] is used to find the best parameters for CAST.

To update particles, we use the most popular method as shown the below.

$$V_{aj}(k+1) = wV_{aj}(k) + c_1 r_{1j}\left(Y_{aj} - X_{aj}\right) + c_2 r_{2j}\left(Y_{gj} - X_{aj}\right),$$
$$X_{aj}(k+1) = X_{aj}(k) + V_{aj}$$

where a is an index to represent a particle in the swarm, j is an index to represent the j^{th} dimensional element of the particle, w is the inertia weight, c_1 and c_2 are the acceleration constants, V_{aj}, Y_{gj}, and Y_{aj} are the velocity, the global best, and the personal best of the j^{th} dimensional element of the a^{th} particle, respectively, and X_{aj} is the j^{th} dimensional element of the a^{th} particle. Using this updating method with the fitness function $\tilde{f}$, the PSO algorithm finds the best parameters.

C. Inverse kinematics

For the haptic guidance controller [10], it is necessary to find the inverse kinematics solutions. The inverse kinematics problem is stated as follows: given a tip position P_{tip}, find four encoder counter values ($\varepsilon_{yaw}, \varepsilon_{ins}, \varepsilon_{pit}, \varepsilon_{rol}$). In this problem (i.e., redundant degrees of freedom), there may be a multiple number of solutions [17]. To solve this problem, we propose an iterative method with geometric and analytical analysis to find all possible solutions.

First, (2) is rewritten as follows:

$$\begin{bmatrix}\Delta x\\ \Delta y\\ \Delta z\end{bmatrix} = R_X(\theta)R_Z(\phi)\begin{bmatrix}L_x\\ L_y\\ L_z\end{bmatrix},\ \begin{bmatrix}L_x\\ L_y\\ L_z\end{bmatrix} = R_X(\psi)R_Z(\alpha)\begin{bmatrix}L\\ 0\\ 0\end{bmatrix} \tag{7}$$

where $[\Delta x \quad \Delta y \quad \Delta z]^T = P_{tip} - P_G$ and P_{tip} is a desired tip position. Δx, Δy, and Δz are expressed as follows:

$$\Delta x = L_x c\theta - L_y s\theta \tag{8}$$

$$\Delta y = L_x c\phi s\theta + L_y c\phi s\theta - L_z s\phi \tag{9}$$

$$\Delta z = L_x s\phi s\theta + L_y s\phi c\theta - L_z c\phi \tag{10}$$

where *cosine* and *sine* are replaced by "c" and "s", respectively. The insertion length (L) and the corresponding encoder counter value (ε_{ins}) are calculated using the equations below.

$$L = \sqrt{(\Delta x)^2 + (\Delta y)^2 + (\Delta z)^2}$$

$$\varepsilon_{ins} = (L - L_{\min}) \cdot \Delta\varepsilon_{ins}/\Delta L \tag{11}$$

where $\Delta L = L_{\max} - L_{\min}$ and $\Delta\varepsilon_{ins} = \varepsilon_{ins,\max} - \varepsilon_{ins,\min}$.

If we use a fixed encoder counter value for the roll (i.e., $\varepsilon_{rol} = constant$), we can find the remaining encounter values. Using (8) with $c^2\theta + s^2\theta = 1$, a quadratic equation is derived as follows:

$$a_\theta s^2\theta + b_\theta s\theta + c_\theta = 1 \tag{12}$$

where $a_\theta = L_x^2 + L_y^2$, $b_\theta = 2L_y \cdot \Delta x$, and $c_\theta = (\Delta x)^2 - L_x^2$. From (12), we have at most two roots in terms of $s\theta$. Because the *sine* function is a periodic function, each root can also map to multiple angles. In CAST, each root can have at most two angles because $|\theta_{\max} - \theta_{\min}| < \pi$. Therefore, there are at most four angles. The corresponding encoder counter values are expressed as follows:

$$\varepsilon_{yaw}^k = (\theta^k - \theta_{\min}) \cdot \Delta\varepsilon_{yaw}/\Delta\theta \tag{13}$$

where $\Delta\theta = \theta_{\max} - \theta_{\min}$, $\Delta\varepsilon_{yaw} = \varepsilon_{yaw,\max} - \varepsilon_{yaw,\min}$, and $1 \le k \le 4$.

Similarly, a quadratic equation is derived using (9), (10), and $c^2\phi + s^2\phi = 1$.

$$a_\phi s^2\phi + b_\phi s\phi + c_\phi = 1 \tag{14}$$

where $a_\phi = (\Delta y)^2 + (\Delta z)^2$, $b_\phi = 2L_z \cdot \Delta y$, and $c_\phi = L_z^2 - (\Delta z)^2$. There are at most 8 angle values due to $|\phi_{\max} - \phi_{\min}| < 2\pi$.

```
ϵ.clear();
For e = ϵ_rol,min : ϵ_rol,max
  ϵ_rol = e;
  Calculate ϵ_ins using (11)
  If ((ϵ_ins ≤ ϵ_ins,min) or (ϵ_ins ≥ ϵ_ins,max))
    continue;
  end
  Find possible θ (at most 4) using (12)
  Find possible ∅ (at most 8) using (14)
  For k = 1: θ.length
    For l = 1: ∅.length
      update ϵ_yaw using θ[k] and (13)
      update ϵ_pit using ∅[l] and (15)
      verify the solution using (7)
      If the solution is valid
        ϵ.push(ϵ_yaw, ϵ_ins, ϵ_pit, ϵ_rol)
      end
    end
  end
end
```

Figure 6. Pseudo code to find all possible inverse kinematics solutions.

The corresponding encoder counter values are expressed as follows:

$$\varepsilon_{pit}^l = (\phi^l - \phi_{\min}) \cdot \Delta\varepsilon_{pit}/\Delta\phi \tag{15}$$

where $\Delta\phi = \phi_{\max} - \phi_{\min}$, $\Delta\varepsilon_{pit} = \varepsilon_{pit,\max} - \varepsilon_{pit,\min}$, and $1 \le l \le 8$. From (13) and (15), there are multiple solution candidates. Using (7) with multiple candidates, the unique solution is selected.

An encoder counter value is an integer value. Also, each encoder has a finite number of encoder counter values. If the entire discrete values of ε_{rol} are considered, all possible inverse kinematics solutions can be explored. Fig. 6 summarizes how to find the entire possible inverse kinematics solutions.

III. Reference Generation for Visual and Haptic Guidance

Using the proposed method as shown in Fig. 6, we can find all possible inverse kinematics solutions. However, this is not suitable for real-time applications because of heavy computational burdens. In order to generate reference encoder counter values for the haptic guidance controller, a simple heuristic is introduced to update ε_{rol} effectively.

A training scenario is designed for a trainee to learn how to move a surgical instrument as shown in Fig. 1 (i.e., a hand-eye coordination task that requires a trainee to maneuver an instrument in order to traverse the recommended path in the 3D working space). For instance, a trainee is asked to touch four targets (e.g., R1, R2, R3, and R4) using the right instrument as a hand-eye coordination task.

While a trainee practices the hand-eye coordination task, CAST provides visual and haptic guidance to him or her. For the visual and haptic guidance, the reference path is required. The path generator [8] generates a shortest and collision-free path given the initial and goal positions. Using this recommended path, several visual cues such as cubes and lines are rendered on a display monitor as visual guidance [9]. Additionally, we can consider an instrument tip's posture (i.e.

orientation) to guide a trainee effectively. (The roll movement determines the posture of the instrument tip.) For instance, first, a trainee is asked to touch the target R1 with a posture A (e.g., an instrument tip facing up) then, the trainee is asked to move to the target R2, and finally is asked to touch R2 with a posture B (e.g., an instrument tip facing the side). In this example, a trainee should change the posture before touching R2. The guidance system provides a recommended posture changing method to a trainee.

To provide this guidance information, we use a path length (PL) and a Euclidean distance ($dist$) between a reference point on the recommended path and the goal position. A nearest point (P_{near}) on the recommended path from the instrument tip is used as a reference point [10]. Given a reference point with a recommended path, the posture guidance information is updated as follows:

$$\varepsilon_{rol} = \varepsilon_{rol,initial} + (\varepsilon_{rol,goal} - \varepsilon_{rol,initial}) \cdot (PL - dist)/PL \quad (16)$$

where $\varepsilon_{rol,initial}$ and $\varepsilon_{rol,goal}$ are the corresponding encoder counter values for roll at the initial posture and the goal posture, respectively.

Using P_{near} and ε_{rol}, the visual guidance cue (i.e., a cube) is updated and the corresponding four encoder counter values are calculated for the haptic guidance controller. In this case, a single ε_{rol} value is used to solve the inverse kinematics problem. Therefore, the heavy computational burdens due to iterations are removed (i.e., the use of inside for-loops in Fig. 6 without iterations). Also, the inverse kinematics solution always exists because the instrument tip traverses in the working space.

IV. Experimental Results

To verify the proposed method, we use the right-hand fixture with Logitech C920 HD pro web camera. The instrument tip position and the HD (1280×720) camera image are updated every 50ms (i.e., 20Hz) to provide the visual and haptic guidance. Also, Eigen [18] – *C++ template library for linear algebra* – is used to implement the proposed method. To render the visual cues and to detect a color marker, the openCV [19] library is used. The camera matrix was calculated using the linear least squares method [7] and we assume that the matrix is reliable to use in CAST. The reliability of the camera matrix was verified using experimental results.

A. Parameter Optimization Results

Data pairs (i.e., four encoder counter values and the actual tip position estimated by color marker detection) are collected by a user. To collect a data set, a user is asked to hold an instrument for a moment (e.g., about 0.5s) while moving the same instrument freely. The collected data are preprocessed to select a stable data set by a classification method (i.e., there are two categories – moving an instrument and pausing an instrument. During the preprocessing, data to represent moving an instrument are filtered out.). The stable data are used as samples to calculate the fitness values for the PSO. For

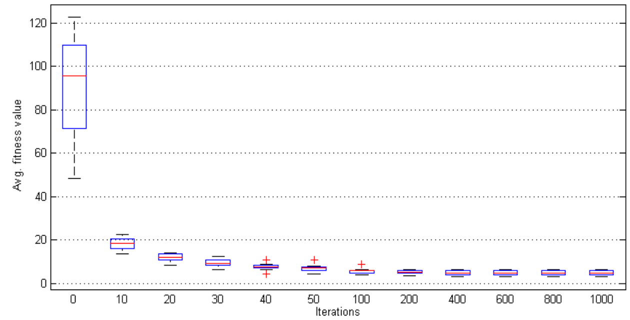


Figure 7. PSO algorithm results (Average fitness values for 10 trials. Each trial has 1000 iterations).

instance, 37 data pairs are used to optimize when we find the parameters for the right instrument.

For the PSO algorithm, 80 particles are used and each particle has 20 parameters as the elements of the particle. To generate initial populations, measurement data are used with reasonable measurement error bounds (e.g., 20° for an angle, 1.5cm for a length, and 50 for encoder counter values). For instance, an insertion length is measured, and the measurement value of $L_{\min}$ is 14cm. The reasonable error bound is 1cm. Therefore, an element for $L_{\min}$ in the initial populations is selected in [13cm, 15cm]. To find the best parameters, w, c_1, and c_2 are set to 0.4, 2, and 2, respectively. Also, PSO repeats at most 1000 times. We run the algorithm 10 times to verify the proposed method (i.e., a single trial contains 1000 iterations.). The PSO results are given in Fig. 7. The average fitness value (i.e., pixel error) is 4.9703 (pixels) after 1000 iterations. It takes about 15 seconds for 1000 iterations. The proposed algorithm minimizes the fitness values rapidly for the first 100 iterations. It means that we can find the reasonable solution for the actual tuning process within 2 seconds. Unlike Fig. 3, the proposed method provides the better visual overlaying performance as shown in Fig. 8. After this optimization process, the color marker is not required for the actual training scenario.

B. Reference Generation Results for theVisual and Haptic Guidance

To illustrate an example of the visual guidance, we use a hand-eye coordination task as presented in Section III. For instance, a user is asked to move a surgical tool from R2 to R3 using the right handed instrument as shown in Fig. 9. While performing this task, the recommended path (green line), the

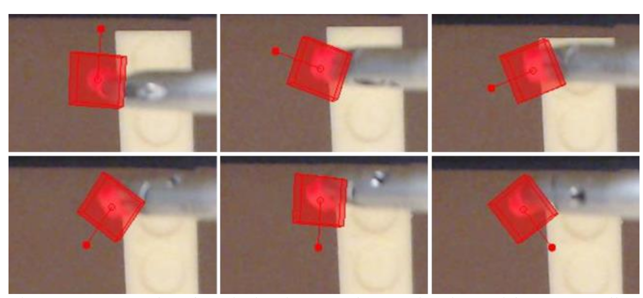


Figure 8. Example of optimization results. Red cubes with a center line are overlaid to represent an instrument tip on the 2D display.

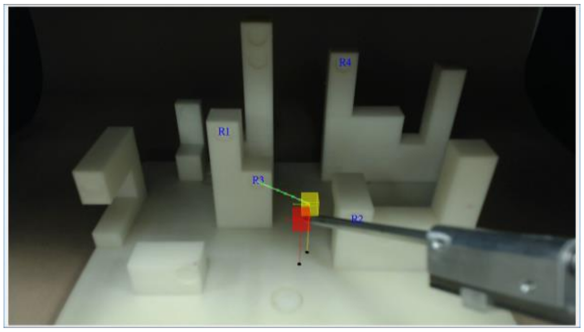


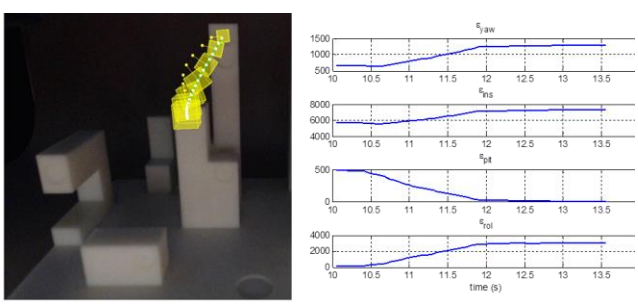
Figure 9. Example of the visual guidance while moving from R2 to R3.

reference point (yellow cube) on the path, the tip position (red cube), and additional visual cues (posts) to indicate a 3D depth are overlaid on the 2D monitor [9]. Using (16), the yellow cube's posture is updated to represent the reference posture. The red cube's posture is also updated to present instrument tip posture by using the live encoder counter values. These visual cues assist a trainee while performing this particular task. For the haptic guidance [10], the encoder counter values at the reference point are updated using inverse kinematics solutions as shown in Fig. 10-(b). The selected reference points on the reference path are illustrated in Fig 10-(a). In this example, the posture changes smoothly because of using (16). If we define another updating procedure, the corresponding visual guidance will assist a trainee.

V. Discussion and conclusion

In this paper, we have presented a mathematical model to solve 4DOF forward and inverse kinematics problems for CAST. The model was derived from the actual movement data. To find the best parameters, the PSO method is used with color marker detection. This is a powerful optimization technique that suits the application on hand well. Our experimental results indicate that the proposed method can apply to real-time applications such as the haptic guidance controller design, which is the key element of our force-based navigation in surgical training [10].

For future work, we will take into account a realistic training scenario, like a peg transfer task. While performing the peg transfer task, a trainee uses a grasper-type instrument to pick a rubber ring up, to carry the object, or to place the object on a peg. To assist a trainee with this task, the proposed position and posture estimation method will be used to support the visual and haptic guidance. The application to other tasks in the fundamentals of laparoscopic surgery training (e.g., cutting or suturing) will be explored as well.



(a) (b)

Figure 10. (a) Example of displaying a reference point and (b) the corresponding encoder counter values for the haptic guidance.

Acknowledgment

This material is based upon work supported by the National Science Foundation under Grant Number 1622589 "Computer Guided Laparoscopy Training". Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

References

[1] Fundamentals of Laparosopic Surgery, http://www.flsprogram.org

[2] Limbs & things, Inc. (LTI), https://www.fls-products.com/home

[3] Simbionix Co., http://www.simbionix.com

[4] Surgical Science, http://www.surgical-science.com

[5] J. Dankelman, "Surgical robots and other training tools in minimally invasive surgery," In Systems, Man, and Cybernetics (SMC), 2004 IEEE International Conference on, pp. 2459 – 2464, 2004.

[6] N. Stylopoulos, S. Cotin, S. K. Maithel, M. Ottensmeyer, P.G. Jackson, R.S. Bardsley, and S.L. Dawson, S. L, "Computer-enhanced laparoscopic training system (CELTS): bridging the gap," Surgical Endoscopy and Other Interventional Techniques, vo. 18, no. 5, pp. 782-789, 2004.

[7] J.W. Rozenblit, C. Feng, M. Riojas, L. Napalkova, A. J. Hamilton, M. Hong, P. Berthet-Rayne, P. Czapiewski, G. Hwang, J. Nikodem, and A. Shankaran, "The computer assisted surgical trainer: design, models, and implementation," In Proceedings of the 2014 Summer Simulation Multiconference, pp. 211-220, 2014.

[8] L. Napalkova, J.W. Rozenblit, G. Hwang, A.J. Hamilton, and L. Suantak, "An Optimal Motion Planning Method for Computer-Assisted Surgical Training," Applied Soft Computing, vol 24, pp. 889-899, 2014.Y.

[9] A. Wagner and J.W. Rozenblit, "Augmented Reality Visual Guidance for Spatial Perception in the Computer Assisted Surgical Trainer," In Proceedings of the 2017 Spring Simulation Multiconference, in press.

[10] M. Hong and J.W. Rozenblit, "A haptic guidance system for Computer-Assisted Surgical Training using virtual fixtures," In Systems, Man, and Cybernetics (SMC), 2016 IEEE International Conference on, pp. 2230-2235, 2016.

[11] M. Hong, J.W. Rozenblit, and A. J. Hamilton, "A Simulation-based Assessment System for Computer Assisted Surgical Trainer," In Proceedings of the 2017 Spring Simulation Multiconference, in press.

[12] A.R. Jordehi and J. Jasni, "Parameter selection in particle swarm optimisation: a survey," Journal of Experimental & Theoretical Artificial Intelligence, vol. 25, no. 4, pp. 527-542, 2013.

[13] R.J. Schalkoff, "Digital image processing and computer vision" vol. 286, New York: Wiley, 1989.

[14] D.A. Forsyth and J. Ponce, "Computer Vision: A Modern Approach," Prentice Hall Professional Technical Reference, 2002.

[15] S. Helwig and R. Wanka, "Particle swarm optimization in high-dimensional bounded search spaces," In Swarm Intelligence Symposium, 2007, SIS 2007, pp. 198-205, 2007.

[16] W.J. Zhang, X.F. Xie, and D.C. Bi, "Handling boundary constraints for numerical optimization by particle swarm flying in periodic search space," In Evolutionary Computation 2004, CEC2004, vol. 2, pp. 2307-2311, 2004.

[17] D. Tolani, A. Goswami, and N.I. Badler, "Real-time inverse kinematics techniques for anthropomorphic limbs," Graphical models, vol. 62, no. 5, pp. 353-388, 2000.

[18] G. Guennebaud, B. Jacob, P. Avery, A. Bachrach, and S. Barthelemy, "Eigen v3", http://eigen.tuxfamily.org, 2010.

[19] G. Bradski and A. Kaehler, "Learning OpenCV: Computer vision with the OpenCV library", O'Reilly Media, Inc., 2008.