Autonomous Bi-Manual Surgical Suturing Based on Skills Learned from Demonstration

Kim Lindberg Schwaner* Iñigo Iturrate† Jakob Kristian Holm Andersen* Pernille Tine Jensen† Thiusius Rajeeth Savarimuthu*

Abstract

We present a novel application of Learning from Demonstration to realize a fully autonomous bi-manual surgical suturing task, including needle pick up, insertion, re-grasping, extraction and hand-over. Surgical action primitives are learned from a single human demonstration and encoded into an action library from which they are pulled to compose more elaborate tasks at planning/execution time. The method is demonstrated in a non-clinical setting, using unmodified surgical instruments with a custom surgical robot system. We use stereo vision to automatically detect the suture needle and entry points to close the control loop and generalize tasks to different task conditions. The suturing task is shown to generalize well to differing initial conditions with a success rate of 17% for the full task, a mean subtask success rate of 75% and mean needle insertion error of 3.3 mm over the course of 46 trial task executions at human speed. Failures could all be attributed to erroneous vision-based detection, pose estimation and robot calibration.

1 Introduction

Suturing is part of many surgical procedures and is a repetitive and time-consuming task to perform. In this paper we present a fully autonomous bi-manual single-throw robotic suturing task based on action primitives learned from human demonstration. Our method works with unmodified surgical instruments and is shown to successfully generalize to differing task conditions with very little human input or parameter tuning required. Our system represents an autonomy level of 2 (task autonomy) according to the framework proposed by [1].

From a single human demonstration we construct a surgical action library of primitives. Each action in the library encodes manipulator motion and a geometric relationship to an object or location in the environment. Manipulator motion is encoded using Dynamic Movement Primitives (DMPs). Actions can be re-used in different environmental conditions and subtasks, can be sequenced to form complex tasks and do not require optimization of task-specific parameters. Additionally, actions learned from demonstration can be used in combination with other methods of motion generation. This paper extends the autonomous needle manipulation task detailed in [2] by fully automating suturing throws using two manipulators simultaneously with a custom surgical robot [3] instead of the da Vinci Research Kit (dVRRK) [4].

A significant amount of the literature on robotic suturing utilizes motion planning for needle insertion, cutting, and grasping [5]–[12]. These techniques rely on optimizing parameters to specific suture placements and needle shape and size, and also for selecting appropriate grasp poses. While such approaches can generate highly tuned motions that ensure optimal task execution, they add complexity to the system, making it more difficult for non-robotics experts, e.g., surgeons, to comprehend and use.

Due to the complexity of fully parameterized planning, a number of approaches have turned to Learning from Demonstration (LfD) in order to simplify the process of task encoding or task learning [13]–[19]. Since Robot-Assisted Minimally Invasive Surgery (RMIS) systems are already teleoperated, existing data from surgeries can be utilized to bootstrap the learning process.

An often seen limitation in surgical task automation is the need for human input during challenging steps. This includes grasping [12], [18], [19], insertion of the needle [14] or registration of target poses in the environment [16]. Thus, many of these approaches are only partially autonomous. Another common limitation is poor generalization to new task condi-
tions; e.g., different suture placement or the movement of target geometries in the environment. Some methods are unable to generalize [13], [17], while others are limited to parameter sets close to the initially taught conditions [15].

Notable challenges specific to automated suturing are detecting the position and orientation of the suture needle and ensuring a stable grasp. Because of these, a number of approaches have proposed using specialized instruments for suturing [20], [21] or custom mechanical modifications to standard surgical instruments to constrain the needle pose [9], [11], [12]. The latter renders the instruments unsuitable for clinical use.

Within LiD, the closest work to our approach is that of [22], who presented an approach to learning from multiple kinesthetic demonstrations using a combination of Dynamic Time Warping, Gaussian Mixture Models and DMPs. Their method is able generalize to changes in task conditions, but control is open-loop. Contrary to them, we learn from a single demonstration that is provided via teleoperation, similar to current RMIS practice. Furthermore, we propose a closed-loop system where feedback from computer vision allows autonomously detection of the needle pose and entry points.

Regarding autonomous suturing, the work by [12] is similar in scope to ours. They proposed a mostly autonomous and closed-loop system using the Raven robot platform [23] and trajectory optimization instead of LiD. To allow needle pose estimation and enhance precision, their work depends on a custom mechanical add-on, with fiducial markers, attached to the grasper jaws of the surgical instrument. This, in turn, requires cameras to face opposite the pointing direction of the surgical instruments and also makes it impossible to locate the suture needle unless it is already held by an instrument. In this paper, we use unmodified surgical instruments to grasp and manipulate the needle, and a stereo camera rig placed in a way that more closely resembles endoscope placement in RMIS.

2 Methodology

In this section we first define the suturing task and give an overview our experimental surgical robot platform. We then establish the methods for constructing a library of surgical action primitives (actions) and how these are used to autonomously perform the suturing task.

2.1 Suturing Task

An overview of the sequence of motions involved in a single-throw suturing task is shown in Fig. 2. Disregarding knot tying, we identify the following subtasks:

I) From an initial state, pick up the needle using the right-hand manipulator (R).
II) Move R to position the needle tip at the desired entry point.
III) Insert the needle using R, initiating the throw.
IV) Extract the needle with the left-hand manipulator (L), completing the throw.
V) Hand over the needle from L to R (subtask V).

In this work we automate all the listed subtasks and show that they can be adapted to differing task conditions.

2.2 Surgical Robot Platform

The system used in this study for demonstration recording, autonomous execution and evaluation is a custom surgical research robot platform [3], which we have aptly named “Modular and Open Platform for Surgical Robotics Research”
MOPS. MOPS is similar in scope and function to the dVRK [4] and Raven-II [23], but can be built from commercially available robot arms. Figure 1 shows an overview of our experimental platform, which consists of the following elements:

- UR5 and UR5e robotic arms (Universal Robots A/S).
- Custom surgical instrument adapters.
- Large Needle Drivers (LNDs) (Intuitive Surgical, Inc).
- Touch haptic devices (3D Systems).
- USB foot pedals.
- Basler acA2500-20gc GigE cameras, each with Tamron M111FM08 8 mm lenses.
- PCs running robot control and computer vision algorithms.
- 3-Dmed soft tissue training suture pad.
- Half-circle 25 mm cutting suture needle.

The suture pad was placed in front of the manipulators on a raised podium that imitates an operating table. We used custommade adapters for actuating surgical instruments and mounting them on the UR robot tool flange. The UR5 robotics arms each have six Degrees of Freedom (DOF) and the instruments each have three DOF, the last being the grasper opening angle. This introduces redundancy in the kinematic chain, which was resolved using a weighted Jacobian pseudo-inverse scheme for solving the inverse velocity kinematics problem. Two Touch haptic devices were used for teleoperating the robot arms. Further details on MOPS, including instrument adapters and appertaining software components, are provided in [3].

The cameras were arranged in a stereo configuration with a baseline of 8.5 cm. Stereo camera position was fixed with respect to the robotic arms and the distance from the focal point to the suture pad was approximately 40 cm. Intrinsic parameters of the cameras were calibrated using the method of [24]. The resulting root mean square re-projection error of calibration board corners in both views was 0.16 pixel in the area of the suture pad.

We obtained the factory calibrations of the UR arms directly from the robot controllers. Calibration of the surgical instrument tip to the UR robot flange was done by manually pivoting the instrument tip around a fixed point and then solving the robot-tool calibration problem using the formulation by [25]. Cameras and robots were calibrated to a common reference frame, $\Phi_{\text{world}}$, using a scheme adopted from [26].

### 2.3 Computer Vision-Based Tracking

Typically, for surgical robots in clinical use, the only readily available sensor feedback is visual. Accordingly, we use stereo vision for detecting locating the suture needle and the entry points marked on the suture pad. Figure 3 shows a frame from the right camera with pose estimation results projected onto it. The instrument Tool Center Point (TCP) frames, $\Phi_{\text{TCP},A}$ and $\Phi_{\text{TCP},B}$, stem from forward kinematics computations and thus hint at the quality of the calibration of the entire system.

The suture pad pose was determined by detecting 3D locations of a number of ArUco markers that were fixed with respect to the suture pad itself. These points were then matched to known points (from physical measurements) defined in the suture pad frame, $\Phi_{\text{pad}}$, using the method of [27], giving the least-squares homogeneous transformation between the camera frame, $\Phi_{\text{optical}}$, and $\Phi_{\text{pad}}$.

We painted the suture needle and needle entry points a distinct color to ease image segmentation, which was done by thresholding in HSV color space. The needle entry points were found by computing the centroids of contours in the binary image, then sorting them geometrically with respect to the detected suture pad frame. The method introduced in [28] was employed to fit ellipses to segmented needle pixels. Pixels within a distance of 3.5 pixel to the ellipse arc in the segmented image were selected as belonging to the needle. 3D points were then reconstructed by triangulation and the sixDOF needle pose of the needle frame, $\Phi_{\text{needle}}$, estimated by first fitting a least-squares 3D plane, $\Omega$, to the points. The plane normal $\mathbf{n}_\Omega$ was then taken as the $z$-axis and the vector $y'$ from the ellipse center to the midpoint of the needle arc, projected onto $\Omega$, was used to compute the $x$-axis as the cross product $x = n_\Omega \times y'$ and finally the $y$-axis as $y = n_\Omega \times x$.

### 2.4 Surgical Action Library

The overarching concept behind our work is to establish what we call a surgical action library, i.e., a collection of self-contained action primitives that can be adapted and re-used in various surgical tasks by sequencing them in different orders and changing the associated target objects.

An overview of the workflow associated with constructing and executing tasks based on the surgical action library is shown in Fig. 4. There are two phases in the workflow: learning and execution. In the learning phase we record tasks demonstrated by a human operator. The recorded data contains the motion of the manipulators and accompanying endoscope video. The demonstrated recordings are segmented, encoded, and stored in the action library. In the action encoding process, a target object frame for the motion is identified. As
an example, for a grasp needle action, the target object frame would be \( \mathcal{F}_{\text{needle}} \), i.e., the movement to pick up the needle is relative to the needle itself. The encoding of physical motion is done using DMPs, as explained in Section 2.5.

In the execution phase, a task planner binds together the primitive actions with the information output from the vision-based tracking system such that tasks may be generalized to the current state of the environment. Notice that many actions are related and/or repeated, and can therefore be re-used under different initial and final conditions by changing parameters such as start- and end-poses. This showcases the reusability that is key to our framework. Similarly, many of the more simple transport motions, e.g., moving to a grasp location, can be easily replaced with an alternative motion planner, while still using actions learned from expert demonstrations for other parts of the task. This showcases the modularity of our approach.

### 2.5 Dynamic Movement Primitives

We use DMPs [29], [30] to encode Cartesian space robot motions. Position and orientation are encoded as follows.

#### 2.5.1 Position

In \( \mathbb{R}^3 \), DMPs represent robot trajectories as a second order dynamical system of the form:

\[
\begin{align*}
\tau^2 \ddot{y} &= a_y (\beta_y (g - y) + \tau \dot{y}) + s_y \mathbf{R} \mathbf{f}(x), \\
\tau \dot{x} &= -a_x x,
\end{align*}
\]

where \( a_y \in \mathbb{R}^+ \) and \( \beta_y \in \mathbb{R}^+ \) are gain constants, \( g \in \mathbb{R}^3 \) is an attractor (goal) point, \( y_0 \in \mathbb{R}^3 \) is the starting point of the system, \( \tau \in \mathbb{R}^+ \) is a time constant corresponding to the duration of the movement, and \( x \) is a phase variable. The system is modulated by a non-linear forcing term, \( \mathbf{f}(x) \), given by Eq. (3), which can be learned to fit any demonstration. Notice that if constants \( a_y \) and \( \beta_y \) are chosen for critical damping, then the system will converge towards \( g \) no matter its starting point \( y_0 \), as the influence of \( \mathbf{f}(x) \) will vanish as \( x \to 0 \).

The forcing term

\[
f(x) = \sum_{i=1}^{N} \psi_i(x) w_i (g - y_0), \quad (3)
\]

\[
\psi_i(x) = \exp \left( -h_i (x - c_i)^2 \right), \quad (4)
\]

consists of \( N \in \mathbb{Z}^+ \) Gaussian basis functions (see Eq. (4)) with centers \( c_i \), widths \( h_i \) and attached weight vectors \( w_i \in \mathbb{R}^3 \).

We apply roto-dilation to the forcing term, similar to [31]. That is, we define \( \mathbf{v} = g - y_0 \) and \( \mathbf{v}_{\text{train}} = g_{\text{train}} - y_{0,\text{train}} \), such that \( s_R = \frac{||\mathbf{v}_{\text{train}}||}{||\mathbf{v}||} \) and that \( \mathbf{R}_g \) is the rotation matrix that rotates from \( \mathbf{v}_{\text{train}} \) onto \( \mathbf{v} \).

As mentioned above, \( f(x) \) can be learned from demonstration. Given a demonstration trajectory \( \{y_{\text{demo}}(t), \dot{y}_{\text{demo}}(t), \ddot{y}_{\text{demo}}(t)\} \), one can calculate

\[
f_{\text{desired}} = \tau^2 \ddot{y}_{\text{demo}} - a_y (\beta_y (g - y_{\text{demo}}) - \tau \dot{y}_{\text{demo}} - \tau \dot{y}_{\text{desired}}). \quad (5)
\]

Since Eq. (3) is linear in its parameters, this can be done by, e.g., least-squares weighted linear regression.

Notice that, if we view Eq. (1) as a spring-damper system with stiffness \( K = \alpha_y \beta_y \) and damping \( D = \beta_y \tau \) will have a significant influence on the system dynamics, as \( K_{\text{effective}} \propto \frac{1}{\tau^2} \) and \( D_{\text{effective}} \propto \frac{1}{\tau} \). In order to avoid this, we recompute \( \alpha_y \) and \( \beta_y \) to ensure that the effective stiffness and damping remain the same under a change of \( \tau \).

#### 2.5.2 Orientation

Equation (1) is applied to learning positions in Cartesian space. For the orientation, a similar formulation can be derived using unit quaternions [32]. Due to space constraints, the formulation is omitted here.

Because of the double-cover \( S^3 \Rightarrow SO(3) \) exhibited by unit quaternions when representing spatial orientations, generalizing the initial and goal orientations of the DMP, \( q_0 \) and \( g_o \), respectively, can result in unexpected behavior if the angular distance between \( q_0 \) and \( g_o \) is vastly different than the training set, i.e., if it rotates “the long way around”. This is due to the forcing term being defined in \( \mathbb{R}^3 \) and thus having no knowledge of the space \( S^3 \). To account for this issue, we make a small addition to Ude et al.’s approach by calculating the angular distance between the initial and goal orientations in the training set

\[
\Theta_{\text{train}} = \left\| 2 \log (g_o, \text{train} \ast \overline{q}_{0,\text{train}}) \right\|, \quad (6)
\]

where \( \ast \) is the quaternion product, \( \overline{q} \) is the quaternion conjugate and \( \log(q) \) is the quaternion logarithmic map \( S^3 \Rightarrow \mathbb{R}^3 \) [32]. This is then compared to the distance between the new initial and goal orientations, with all possible sign combina-

---

1Note that for this particular suturing task we chose to use roto-dilation only for suturing subtasks III and IV, cf. Section 2.1. For all other actions it was disabled by setting \( s_R = 1 \) and \( \mathbf{R}_g = I_3 \) in Eq. (1).
tions of $q_0$ and $g_o$, i.e.,

$$\theta_1 = \|2 \log (g_o * \overline{q_0})\|, \quad (7)$$
$$\theta_2 = \|2 \log (-g_o * \overline{q_0})\|, \quad (8)$$
$$\theta_3 = \|2 \log (g_o * -\overline{q_0})\|, \quad (9)$$
$$\theta_3 = \|2 \log (-g_o * -\overline{q_0})\|. \quad (10)$$

and choosing the one that minimizes the angular distance $|\theta_{train} - \theta|$.

3 Experimental Evaluation

In this section we document experiments where a single-throw suturing task was learned from a single demonstration, using the methods detailed in Section 2, and then subsequently executed autonomously with initial task conditions different from those of original demonstration.

3.1 Task Demonstration

To learn and encode the actions involved in a suturing task, we first demonstrate it by teleoperating two LNDs using the system shown in Fig. 1 and further detailed in Section 2.2. The human operator demonstrated the task as described in Section 2.1, and shown in Fig. 2. The following signals were recorded:

- Six-DOF poses of the end-effectors of the LNDs, $\mathcal{F}_{TCP,A}$ and $\mathcal{F}_{TCP,B}$ in the base frame of the corresponding robot.
- Opening angles of the LND grasper jaws.
- Synchronized frames from both cameras.

- Vision-based tracking results (with respect to camera frame, $\mathcal{F}_{optical}$):
  - Six-DOF poses of the suture pad, $\mathcal{F}_{pad}$.
  - Six-DOF needle poses, $\mathcal{F}_{needle}$.
  - Needle insertion point positions, $\mathcal{F}_{entry,0,1,2}$.

The demonstration data was manually segmented into a series of primitives by examining zero or near-zero velocity crossings, as is common in LiD [33]. The manipulator motion of each primitive was encoded as a DMP, as described in Section 2.5. The combined action, consisting of the DMP and associated target object was then saved to the surgical action library.

Note that the number of segmented action primitives was larger than the number of subtasks listed in Section 2.1 as the subtasks were further divided to obtain better action reusability. For instance, the needle pick-up subtask (I) was further divided into the actions move to grasp pose ($B$); close needle grasper jaws ($B$); and pull out needle backward ($B$). Similarly, the needle hand-over subtask (V) was divided in position needle in hand-over location ($A$); move to grasp pose ($B$); close grasper jaws ($B$); and release grasper jaws ($A$).

3.2 Task Execution

Action primitives were pulled from the previously constructed action library to sequence a single-throw suturing task, following the subtasks described in Section 2.1. The vision system of Section 2.3 was used to continuously estimate the six-DOF needle pose and to detect the location of the entry points on the suture pad. This information was then used to generalize actions to the current conditions of each trial.

The suturing task was executed in a total number of 46 trials.
Throughout the trials, the initial conditions were changed to evaluate how well our approach generalizes. Figure 6 shows some examples of how the initial conditions were changed and Table 1 summarizes the mean and lower and upper bounds of the variation among all trials. The execution speed was the same as that of the demonstration.

We noted the successful completion of the following key actions in each trial: Needle pick-up (B); needle insertion (B); needle re-grasping (A); needle extraction (A); and needle hand-over (A to B). In our experiments, the failure of any given key action would halt execution and cause all subsequent actions to automatically fail. Figure 7 plots the percentage of successful completions of each key action against both the total number of trials and against the number of executions of that individual action.

Referring to the plot in Fig. 7, the success rate for the full task (as defined in Section 2.1) was 17% and the mean success rate for the total number of individual key actions was 75%. The initial needle pick-up was done successfully in 98% of all trials (45 out of 46). Needle insertion was successful in 96% of all trials (44 out of 46). Needle re-grasping was less successful with 59% (27 of 46). The total number of successful needle extraction actions was lower still with 48% (22 out of 46), but when we discount cases where a failed re-grasp would automatically cause the subsequent extraction to fail, the success rate was 81%. Needle hand-over was the least successfully completed action, succeeding in 17% of all trials (8 of 46), but when accounting for the failure of a previous action, which would cause the hand-over to automatically fail, the action was completed successfully 36% of the time.

Besides key action completions, we also measured the needle insertion error, $e_{in}$, which we define as the Euclidean distance from the desired entry point, $p^{\text{entry,desired}}$, to the point on the tissue pad surface where the needle tip was actually inserted, $p^{\text{entry,actual}}$, i.e.,

$$
e_{in} = \|p^{\text{entry,desired}} - p^{\text{entry,actual}}\|.
$$

\begin{table}[h]
\centering
\begin{tabular}{|c|c|c|c|c|c|c|}
\hline
Object    & Translation [mm] & Rotation [°] \\
\hline
Suture pad & -7.14±2 & 24.6±59 & 6.0±23 & 11.1±40 \\
Needle & -17.79±43 & -3.25±24 & -5.30±38 \\
\hline
\end{tabular}
\caption{Variation of initial conditions relative to the original demonstration. The notation $x_{\pm \delta}$ expresses the mean and the lower and upper bounds of the variation throughout all trials. Variations in orientation are expressed by a rotation angle $\theta$ about an arbitrary axis. Note that needle entry points were marked on the suture pad and thus were moved correspondingly.}
\end{table}
Figure 8 plots the needle insertion error grouped by entry point. The mean error at entry point 1 was 4.3 mm, at point 2 it was 3.2 mm and at point 3 it was 3.5 mm. The overall mean insertion error was 3.3 mm. Note that, although entry point 1 was used in the original demonstration, it was located differently in each trial as the whole suture pad was positioned in different locations and orientations over the course of all trials.

4 Discussion

The experiments documented in this paper confirm that the proposed method can be used to autonomously perform a single-throw suturing task, using unmodified surgical instruments, while adapting to changes in the task conditions, although not without problems.

The largest number of failures happened when attempting to re-grasp the needle after it had first been inserted, and while handing over the needle from arm $A$ to $B$. Figure 9 shows some examples of failure modes. The main reason was erroneous vision-based detection and pose estimation of the suture needle. These errors could in many cases be attributed to the HSV color segmentation method, which was very sensitive to changes in lighting conditions. While the needle was inserted or held by the grasper jaws it became partly occluded or shadowed by the instruments themselves, which also worsened pose estimates. As our system currently relies on only visual feedback for closing the control loop this represents a single point of failure.

Another known source of error was that the instrument adapter of arm $A$ had been damaged at the time of execution. This let the instrument move slightly in the mounting bracket, causing the robot-to-instrument calibration to be invalidated over time as the environment exerted forces on the instrument. This is apparent in Fig. 3, where $\mathcal{F}_{TCP,A}$ is offset by some millimeters from where it should be: at the point between the tips of the grasper jaws.

As is evident from Table 1 and Fig. 6, the initial task conditions were varied throughout the trials. In general, we found that the system handled these changes successfully, i.e., changing the initial conditions of the task did not significantly impact its successful completion. There was one notable exception to this, however: Changing the suture pad position or orientation would easily render entry points impossible to see from either the left or right frames because of the large baseline distance between the cameras. This, in turn, made estimating entry point positions hard or entirely impossible. Making it possible for the camera to be moved by placing it on a third robot arm could mitigate this issue.

The needle insertion error plot in Fig. 8 shows that, in most cases, the needle was inserted a distance of 5 mm or less from the desired point. The mean error distance for all entry points combined was 3.3 mm. Whether this is acceptable or not is highly dependent on what part of the anatomy is being sutured. A worst-case error of around 6.5 mm with one outlier at around 10 mm, a wrongly detected insertion point, shows that our system is not completely reliable.

Compared with our previous work [2], we successfully perform complete suturing throws, and not just needle pick-up and partial insertion. Furthermore, we use two manipulators of a custom surgical robot as opposed to a single manipulator of the dVRK, which also shows that our methods work with more than one type of system. We observed similar needle insertion errors (3.3 mm now, 3.8 mm previously), although in this paper we performed many more trials (46 now, 16 previously), lending more statistical power to our results.

In comparison with [12], the mean needle insertion error of our system is greater (3.3 mm for ours, 0.71 mm for theirs). They report 16 successful trials, but no failures. Based on that information their system is more reliable. This is not surprising, seeing as their mechanical needle guide and fiducial markers eliminate a lot of uncertainty in the needle pose estimate, which is one of our primary issues. However, their mechanical needle guide also prevents locating and picking up the needle from an unknown location, and the fiducials have to always be visible in the camera view for their method to work. Another notable difference is that we are able execute the task at human speed, whereas theirs is slower (the exact speed is not documented in the paper). We believe our work is slightly closer to realistic clinical application because we do not require modifications to the surgical instrument or cameras facing opposite the insertion direction of the instruments.

In our experiments, we did not yet attempt faster-than-human execution speeds although, in principle, there is nothing preventing this. We did endeavor to perform multiple suture throws by sequencing the appropriate actions from the ac-
tion library. However, in practice this was unsuccessful because of the aforementioned reasons for task failure. A suture thread was not added, as we did not have a mechanism in place to handle it, to avoid further complicating the task.

Future work is planned to improve the detection and tracking of the system by replacing the color-based image segmentation method with a convolutional neural network. A better segmentation result will allow for more accurate pose estimation. Pose estimation will be further improved by including forward kinematics as sensor input for needle tracking, thus mitigating issues with occlusion. Finally, we will investigate automating more elaborate and different surgical tasks.

5 Conclusion

In this paper we demonstrated a fully autonomous single-throw suturing task which generalized to differing task conditions. We used a custom robot system for both human demonstration and autonomous execution. The task was composed by actions from a surgical action library, where actions were learned from a single human demonstration and stored for later execution. The success rate for the full task was 17%. The success rate for individual key actions of the suturing task ranged from 98% for needle pick-up and 96% for needle insertion, 81% for needle extraction to 60% for needle re-grasping (following insertion) and 36% for needle hand-over. Failures were caused by needle pose estimation and calibration errors. A mean needle insertion error of 3.3 mm and worst-case errors typically around 6.5 mm show reasonably accuracy for the system, but may not be good enough for realistic clinical application.

Our method can potentially relieve the surgeon of having to perform routine tasks, freeing up valuable resources. To stay close to realistic clinical conditions, we avoid modifications to surgical instruments and use only the joint encoders of the robot and a stereo camera for sensor input. Furthermore, our system requires very little human input or parameter tuning, as opposed to more complex task-specific planning methods. Future work is planned to improve the components which were identified as the main causes for failure during autonomous task execution.

Acknowledgment

We thank Christian R. Dam for his work on the adapters for surgical instruments and Miha Deniša for his work on system calibration. We also thank Michael K. Schmidt and Nicolai Iversen for their help in setting up and repairing parts of our robot system.

References


