Autonomous Robotic Suction to Clear the Surgical Field for Hemostasis using Image-based Blood Flow Detection

Florian Richter\textsuperscript{1} Student Member, IEEE, Shihao Shen\textsuperscript{1} Student Member, IEEE, Fei Liu\textsuperscript{1}, Jingbin Huang\textsuperscript{1}, Emily K. Funk\textsuperscript{2}, Ryan K. Orosco\textsuperscript{2} Member, IEEE, and Michael C. Yip\textsuperscript{1} Member, IEEE

Abstract—Autonomous robotic surgery has seen significant progression over the last decade with the aims of reducing surgeon fatigue, improving procedural consistency, and perhaps one day take over surgery itself. However, automation has not been applied to the critical surgical task of controlling tissue and blood vessel bleeding—known as hemostasis. The task of hemostasis covers a spectrum of bleeding sources and a range of blood velocity, trajectory, and volume. In an extreme case, an un-controlled blood vessel fills the surgical field with flowing blood. In this work, we present the first, automated solution for hemostasis through development of a novel probabilistic blood flow detection algorithm and a trajectory generation technique that guides autonomous suction tools towards pooling blood. The blood flow detection algorithm is tested in both simulated scenes and in a real-life trauma scenario involving a hemorrhage that occurred during thyroidectomy. The complete solution is tested in a physical lab setting with the da Vinci Research Kit (dVRK) and a simulated surgical cavity for blood to flow through. The results show that our automated solution has accurate detection, a fast reaction time, and effective removal of the flowing blood. Therefore, the proposed methods are powerful tools to clearing the surgical field which can be followed by either a surgeon or future robotic automation developments to close the vessel rupture.

I. INTRODUCTION

Since the deployment of surgical robotic devices such as the da Vinci\textsuperscript{®} Surgical System, efforts to automate surgical tasks have become a popular area of research [1]. The automation of surgery has promised to help reduce surgeon fatigue and improve the procedural consistency between surgeries, and perhaps one day take over the surgery itself. Success in realizing surgical automation has accelerated in recent years, with improvements in available open-source platforms such as the da Vinci Research Toolkit (dVRK) [2] and simulators [3], coupled with significant advances to data-driven controller synthesis. Successful demonstrations of automated tasks have included cutting [4], [5], suture needle hand-off [6], suturing [7], [8], and debridement removal [9]. Recent developments in perception for surgical robotics helps bridge these autonomous policies in ideal scenarios to realistic, deformable and noisy tissue environments such as the SuPer frameworks [10], [11].

While progress in developing autonomous surgical tasks has leapt forward, a key area that has been given little attention are reactive policies to traumatic events, such as hemostasis. Hemostasis describes a state of the surgical field that is fulfilled when there is no site of active bleeding and the tissues are unobstructed by blood. The bleeding can originate from a visible or macroscopic blood vessel (artery or vein), or from the microvasculature and capillary network within tissues. Unlike previously automated tasks that occur in a more predictable cadence within a procedure, bleeding can be unpredictable which necessitates hemostasis maneuvers at any time during any surgery. Surgical manipulation of any type—suction, grasping, retraction, cutting, dissection—can immediately cause bleeding. Bleeding can also occur in a delayed manner, for example if a vessel is
not definitively sealed, it can rupture spontaneously without direct contact. This work specifically addresses the problem of small/medium vessel rupture. Overall, there are four distinct stages in hemostasis of this scenario: (1) clearing the surgical field of blood; (2) identification of the bleeding source (vessel rupture); (3) grasping the identified vessel to temporarily stop bleeding; (4) closing the vessel rupture, usually with an energy device, clip, or suture. Each stage requires a complex set of manipulation skills as well as perception algorithms that make it non-trivial to implement.

In this paper, we describe an automated solution to the first task, clearing the surgical field, as shown in Fig. [1]. This task involves first recognizing blood in a scene, then tracking blood flow temporally, and finally prescribing a real-time trajectory generation strategy that will intelligently control a suction tool to siphon the blood to efficiently suction it. To this end, we present the following novel contributions:

1) the first complete automated solution for clearing the surgical field of flowing blood from a ruptured vessel using a robotic suction tool,
2) a novel blood flow detection and tracking method by utilizing temporal visual information, and
3) a trajectory generation technique from blood regions in the image frame for a surgical suction tool to follow and clear the surgical field.

The blood flow detection and tracking method is tested within various simulated scenes as well as an real-life case involving a vessel rupture during thyroidectomy. The complete solution is tested in a lab setting with the da Vinci Research Kit (dVRK) [2] and a simulated surgical cavity for blood to flow through and collect in. The results from the experiments show the effectiveness of the blood flow tracking and surgical suction tool trajectory generation developed in this work.

### II. RELATED WORKS

Previous work on blood detection largely is from the context of Wireless Capsule Endoscopy (WCE) where image processing for detections is used to speed up clinician workflow [12]. The typical approach to blood detection in WCE is to classify either on a pixel level or using patch-based methods [13]. The feature space used for classification is either direct Red, Green, Blue (RGB) [14] channels or the transformed Hue, Saturation, Value (HSV) channels [15]. To efficiently process these color spaces, techniques such as support vector machines [16], chromium moments combined with binary pattern textures [17], and neural networks [18], [19] have been demonstrated. While these methods are robust to small individual lesions, in a surgical scene there can be stains from previous ruptures and larger amounts of blood flow that make the problem of blood detection and, specifically, tracking, a more challenging and complex problem.

There has been previous research on robots interacting with liquids in the act of pouring [20]. However, these methods cannot be applied to a surgical setting since they are limited to specific objects for pouring and capturing liquids. Schenck and Fox applied deep neural networks to detect fluids [21] that can be combined with differential fluid dynamics to reconstruct 3D information [22]. This detection method however requires labelled real-world data which are challenging to collect in a surgical context. Yamaguchi and Atkeson instead used the heuristic of optical flow to detect moving fluid [23]. The work presented here also uses optical flow to detect blood flow. However, instead of the classical method used by Yamaguchi and Atkeson, we applied a deep learning technique for improved performance of optical flow estimation in a surgical environment and fused the detections temporally with a novel temporal filter.

### III. METHODS

An overview of the proposed algorithms for blood region estimation and trajectory generation for an autonomous suction tool on a surgical robot is provided in Algorithm 1.

![Algorithm 1: Blood Suction](image)

At a high level, the blood region is estimated by updating a probability map on the scene, which describes the probability of each pixel in the image frame being blood or not. From the probability map, the blood region is extracted. A trajectory is then generated for the suction tool to follow in order to clear the surgical field from blood. The trajectory is constrained to stay within the blood region to maximize the blood removed.
which describes an observation model for the hidden state $p_t \in \{b, \overline{b}\}$.

### B. Temporal Filtering for Blood Region Detection

Although the magnitudes of optical flow provide a good initial estimate for blood detection, they are nevertheless noisy and require filtering. Therefore, a temporal filter is based on a Hidden Markov Model (HMM) is proposed to fuse independent measurements of the pixel labels over time. The HMM tracks the discrete states for $p_t$ using the observation models in (1). Let the following be a transition probability for a pixel $p_t$ be

$$P(p_{t+1} = b \mid p_t = b), \quad (2)$$

which models the probability that if a pixel is already blood it will continue being blood. In the case of blood vessel ruptures, this should be set close to 1 since the vessel rupture will not stop emitting blood until it has been closed. For the transition probabilities where the pixel is not blood at time $t$, an additional parameter, $k_t^p$, is introduced to the model:

$$P(p_{t+1} = b \mid p_t = \overline{b}, k_t^p = b)$$  \hspace{1cm} (3)

$$P(p_{t+1} = b \mid p_t = \overline{b}, k_t^p = \overline{b}), \quad (4)$$

where $k_t^p$ describes the state of the neighboring pixels of $p_t$. This is modelled as the resulting Boolean-OR operation ($\lor$) on the states of the neighboring pixels:

$$k_t^p = \bigvee_{q_t^i \in A_p} q_t^i \quad (5)$$

where $A_p$ is the set of neighboring pixels to $p_t$. Therefore, the model from (3) is capturing the flow of the blood and (4) is describing the probability a blood source starts at pixel $p_t$.

The temporal filter is designed to estimate the posterior probability of the state $p_t$ using transition probabilities and observation models. This is done using a predict and update step after every detection. The predict step can be calculated as:

$$P(p_{t+1} \mid z_{1:t}^p) = P(p_{t+1} \mid p_t = b)P(p_t = b \mid z_{1:t}^p) + \sum_{k_t^p \in \{b, \overline{b}\}} P(p_{t+1} \mid p_t = \overline{b}, k_t^p)P(p_t = \overline{b}, k_t^p \mid z_{1:t}^p) \quad (6)$$

and the update step is computed:

$$P(p_{t+1} \mid z_{1:t+1}^p) \propto P(z_{t+1} \mid p_{t+1})P(p_{t+1} \mid z_{1:t}^p) \quad (7)$$

However, the predict expression has the joint probability of $p_t$ and $k_t^p$. Explicit estimation for this joint probability would be computationally intractable, so each pixel’s probability of being blood is approximated to be independent of all others at time $t$. With this simplification, the predict step can be rewritten as:

$$P(p_{t+1} \mid z_{1:t}^p) = P(p_{t+1} \mid p_t = b)P(p_t = b \mid z_{1:t}) + \sum_{k_t^p \in \{b, \overline{b}\}} P(p_{t+1} \mid p_t = \overline{b}, k_t^p)P(k_t^p \mid z_{1:t}^p)P(p_t = \overline{b} \mid z_{1:t}^p) \quad (8)$$

### A. Detecting Flowing Blood in Image Frame

Optical flow is chosen to detect flowing blood because it extracts information about all moving objects in the scene. In a surgical scene, the main motion comes from surgical tools and flowing fluids. Another source of motion in robotic surgery comes from a moving camera, but for robotic surgery the camera remains stationary when work is being done in a scene and its position is reset only to change the field of view. We therefore consider only stationary scenes. To mask instrument motion from the scene, previously developed methods can be applied to effectively segment and remove pixels attributed to surgical tools from image [10], [11], [24].

To estimate optical flow, a pretrained convolutional neural network (CNN) is used [25]. A deep learning strategy is used instead of traditional methods such as Lucas-Kanade [26] (used in previous work in robot pouring [23]) as traditional optical flow approaches utilize brightness constancy constraint assumption, and this assumption is not valid in endoscopic procedures due to irregular lighting. Meanwhile, the proposed architecture by Teney and Herbert [25] is able to extract motion from learned features that are invariant to textures, brightness, and contrast, which is ideal for detecting flowing blood from an endoscope.

Similar to previous work in robot pouring [23], we also found experimentally that the magnitude of optical flow is a good signal for detecting fluid motion while the orientation is not. An example of the processed image is shown in Fig. 2 comparing the RGB image to the amplitude map for optical flow.

Consider the magnitude of optical flow at pixel $p$. Let $z_t^p$ be the random variable describing the detection of blood at pixel $p_t$ at time $t$. The detection is modelled where blood is detected, $z_t^p = b$, if the magnitude of optical flow at pixel $p_t$ is greater than a threshold, $\gamma_\alpha$. The inverse is also set, so no blood is detected, $z_t^p = \overline{b}$, if the magnitude at pixel $p_t$ is less than $\gamma_\alpha$. Hence the probability model for these detections can be simply written as:

$$P(z_t^p = b \mid p_t = b) \quad P(z_t^p = b \mid p_t = \overline{b}) \quad (1)$$

Fig. 2: The above images are generated by optical flow estimation from an in-vivo surgical scene. The left image shows the vectors of estimated image motion from optical flow, and the right image is a normalized heatmap of their magnitudes. Notice that the magnitude of optical flow detects the regions of blood flow well, while the orientation gives inconsistent information about the flow.
Algorithm 2: Trajectory Generation for Suction

Input: Start pixel, \( s_t \), end pixel, \( e_t \), and mask of blood region \( B_t \)

Output: Trajectory \( T_t \)

1. Initialize clearance reward \( R \leftarrow 0 \)
2. Initialize temporary eroded blood region \( E \leftarrow B_t \)
3. Initialize \( i \leftarrow 0 \)
4. while \( E \neq 0 \) and \( i < \gamma_r \) do
5. \( E \leftarrow \text{erode}(E) \)
6. \( R \leftarrow R + rE \)
7. \( i \leftarrow i + 1 \)
8. Initialize pixels cost to go to goal: \( D \leftarrow \infty \)
9. Initialize visited map: \( V \leftarrow !B_t \)
10. Initialize parents of nodes: \( K \leftarrow \text{UNDEFINED} \)
11. \( D(s_t) \leftarrow 0 \)
12. while \( V \neq 1 \) do
13. \( u \leftarrow \text{argmin}_{\text{pixel}} D \)
14. \( V(u) \leftarrow 1 \)
15. if \( u = e_t \) then
16. break
17. for pixel \( q \) neighboring \( u \) and \( V(q) \neq 1 \) do
18. if \( D(q) > ||q - u|| - R(q) \) then
19. \( D(q) \leftarrow ||q - u|| - R(q) \)
20. \( K(q) \leftarrow u \)
21. Initialize trajectory for output: \( T_t \leftarrow [ \ ] \)
22. Initialize parent traversal node: \( u \leftarrow e_t \)
23. while \( u \neq s_t \) do
24. Insert \( u \) at beginning of \( T_t \)
25. \( u \leftarrow K(p) \)
26. return \( T_t \)

and an expression can be found for \( P(k_t^p \mid z_{1:t}^p) \) using the inclusion-exclusion principle:

\[
P(k_t^p \mid z_{1:t}^p) = \sum_{j=1}^{|A_p|} (-1)^{j-1} \sum_{J \subseteq A_p, |J| = j} \prod_{q' \in J} P(q_t^q \mid z_{1:t}^p) \tag{9}
\]

This results in the ability to compute and track the probabilities of each pixel being blood using [8] and [7] after every detection.

To find the region of blood on the image frame, a mask is generated where all pixels with a posterior probability greater than 0.5 is set to 1, and the rest are set to 0. Then dilation and erosion morphological operations are applied once to reduce noise on the mask. Finally, the largest connected region of the mask is considered the region with blood flowing if its size is greater than a threshold of \( \gamma_B \). This threshold keeps a detection from occurring when there is no actual blood flowing.

C. Trajectory Generation for Blood Suction

A start and end point must be decided to generate a trajectory for suction. The end point should be roughly near the location of the vessel rupture in order to continuously remove any newly released blood. Meanwhile, the starting point should be downstream of the flowing blood in order to effectively clear the surgical field when suctioning upstream towards the source. Therefore, a simple estimation for the start and end point is done based on the age of the pixels in the blood region. The pixel with the largest and smallest ages in the blood region are defined as the end and start points respectively. To ensure that the end point is not generated at the exact edge of the blood stream, the blood region is eroded before selecting it.

The trajectory generated from the start to end point should also maximize its ability to suction blood while moving upstream. Therefore, using standard minimum distance paths are not ideal as they would tend to plan towards the edges of the blood region rather than the center. To center the trajectory in the blood region, an additional clearance reward is given to the motion planner. The clearance reward is generated by iteratively eroding the blood region for a max of \( \gamma_r \) iterations. The pixels left in the eroded region are given an additional reward of \( r \) at each iteration. The final trajec-

Fig. 3: Example of trajectory generation for blood suction from in-vivo scene. In the top figure, the white highlighted region is the detected blood region and the blue path is the generated trajectory. The starting and stopping points are chosen from the newest and oldest regions of the detected blood respectively, and a heatmap of the normalized detected blood age is shown in the bottom left figure. The generated path uses an additional clearance reward, which is shown in the bottom right figure, to reward paths that stay centered along the blood stream.
B. Datasets to Evaluate Blood Region Detection

Two separate datasets were generated for this work to evaluate the proposed blood region detection algorithm. Both datasets have labelled ground-truth masks, $G_t$, of the blood region. Performance is evaluated from these datasets using the Intersection over Union (IoU) metric:

$$\text{IoU} = \frac{B_t \cap G_t}{B_t \cup G_t}$$

where $B_t$ is a mask of the detected blood region from our proposed method, $\cap$ is the Boolean-AND operation and $\cup$ is the Boolean-OR operation.

1) Simulated Scenes: Six simulated scenes of flowing blood are generated using Unity3D’s particle-based fluid dynamics (PBDs). The scenes are shown in Fig. 4. A total of 61 image frames were extracted per scene. The ground-truth mask, $G_t$, of the blood region is generated by projecting the fluid particles onto a virtual camera’s image plane and applying Gaussian smoothing. The rendered image is set to $1095 \times 1284$ pixels.

A. Implementation Details

All subsequent experiments were ran on a computer with Intel® Core™ i9-7940X Processor and NVIDIA’s GeForce RTX 2080. The blood flow detection and trajectory generation algorithms were implemented in MATLAB. The CNN for optical flow estimation [25] is pre-trained on the Middlebury dataset [27] and uses $l = 3$ image frames for input. The resolution of the optical flow estimation is $1/4$ of the input frame resolution. The size of the probability map is set to the optical flow resolution. The threshold for detection, $\gamma_d$, region size, $\gamma_r$, maximum number of erosions for clearance, $\gamma_c$, and trajectory length, $\gamma_T$, are set to 0.45, 20, 4, and 30 respectively. The detection probability, $P(z_t^p = b | p_t = b)$, $P(z_t^p = b | p_t = \overline{b})$, are set to 0.95 and 0.2 respectively. The initial probability of a pixel being blood, $P(p_0 = b)$ and transition probabilities of a pixel being blood, $P(p_{t+1} = b | p_t = b)$, $P(p_{t+1} = b | p_t = \overline{b}, k_t^p = b)$, $P(p_{t+1} = b | p_t = \overline{b}, k_t^p = \overline{b})$, are set to 0.1, 0.98, 0.85, and 0.01 respectively. The neighbors for a pixel, $A_p$, are set to just up, down, left, and right. The clearance reward per erosion, $r$, is set to 0.2.
Fig. 6: Example figures of blood flow detection and tracking. The top and bottom sequences are from an in-vivo rupture and simulated scene respectively. The highlighted green regions are the intersection between ground truth and blood flow detection. Meanwhile, the highlighted blue regions are the error of detection, i.e., the union minus intersection between the ground truth and blood flow detection. These figures are best viewed in color.

2) In-Vivo Surgical Scene: After the completion of a thyroidectomy conducted on a pig (UCSD IACUC S19130), a rupture occurred on the carotid artery. The video data from this incident is used to evaluate the blood flow detection and tracking algorithm in a similar manner to the simulated scenes. For ground-truth masks of the blood region, $G_t$, 10 out of a total of 41 frames were manually annotated. The recorded image size was 480 by 640 pixels.

C. Performance of Blood Region Detection

To show the effectiveness of the tracking algorithm, a comparison experiment was conducted where the blood flow region was simply set to be the pixels with optical flow magnitude greater than $\gamma_O$. The distribution of IoU results are shown in Fig. 5 for blood flow region detection with and without the tracking algorithm on the simulated scenes and in-vivo dataset. There is a clear difference in performance of the blood flow detection and tracking between the simulated scenes and in-vivo rupture. We believe this is due to the poorer lighting conditions and the reflective surgical clamps used in the in-vivo scene as seen in Fig. 6. Nonetheless, the blood region is successfully estimated when using the tracking algorithm despite the many red stains, hence highlighting the importance of using temporal information for detection rather than color features. For additional comparison, Lucas-Kanade’s [26] and Farnebeck’s [28] optical flow estimation techniques were replaced for the CNN based optical flow estimation [25]. Note that Lucas-Kanade’s optical flow estimation is the proposed detection method for fluids by Yamaguchi and Atkeson used for robot pouring [23]. The resulting IoU for the in-vivo and simulated scenes was measured to be consistently below 0.50 in both cases, which is substantially lower than our proposed detection technique.

Fig. 7: Endoscopic view of phantom used for the automated suction experiments. The red arrows highlight the direction of flow at the four injection points tested in the experiment.

D. Automated Suction in Cavity

To evaluate the complete autonomous surgical task of recognizing blood flow and performing autonomous suction, a tissue phantom with a cavity for liquid to flow through is constructed from silicone, and water with red coloring dye is drained into the cavity using surgical tubing as shown in Fig. 7. A stereoscopic camera with a resolution of $1080 \times 1920$ pixels at 30fps on the dVRK [2] is used. The trajectory generated for suction is converted into 3D position commands using the Pyramid Stereo Matching Network (PSMNet) [29], which takes the stereo-images of the cameras and determines the depth of each pixel. PSMNet’s weight are provided by their original implementations without any task-specific fine-tuning; the maximum disparity is set to 192. PSMNet estimated depth using an image size of 480 by 680 pixels meanwhile the blood flow detection algorithm used
a reduced image size of 120 by 160 pixels to improve its speed.

A Patient Side Manipulator (PSM) from the dVRK [2] was equipped with a da Vinci® Si Suction Tool and followed the generated trajectory to clear the simulated surgical cavity from blood. To follow the trajectory, the position of the end-effector in the PSM base frame, \( \mathbf{b}_s \), is iteratively set to:

\[
\mathbf{b}_{t+1} = \begin{cases} 
\gamma_s \frac{\mathbf{d}_t}{\|\mathbf{d}_t\|} + \mathbf{b}_t & \text{if } \mathbf{d}_t > \gamma_s \\
\mathbf{d}_t + \mathbf{b}_t & \text{if } \mathbf{d}_t \leq \gamma_s
\end{cases}
\]

(11)

where \( \gamma_s = 0.75 \text{mm} \) is the max step size, the operator \( \gamma = [\gamma_x \gamma_y \gamma_z] \) gives the homogeneous representation of a point, and the direction, \( \mathbf{d}_t \), is computed as

\[
\mathbf{d}_t = \mathbf{T}_{\mathbf{b}^s}^T \mathbf{b}^g - \mathbf{b}_t
\]

(12)

The camera to base transform, \( \mathbf{T}_{\mathbf{b}^s}^\mathbf{b}\in SE(3) \), is estimated in real time using our previous work [10] and \( \mathbf{b}^g \) is the 3D goal position generated by the trajectory and PSMNet. This controller is ran at 100Hz and is repeated until \( \|\mathbf{d}_t\| < 2\text{mm} \) per target position, \( \mathbf{b}^g \), from the generated trajectory. Meanwhile the orientation of the suction tip is set to always be in direction of gravity. The position, \( \mathbf{b}_s \), and orientation of the end-effector is converted to joint angles using an analytical inverse kinematic solution. These joint angles are then regulated using dVRK [2].

To account for imperfections in the 3D depth estimation from PSMNet and surgical tool tracking to regulate the end-effector, the suction tool was commanded to oscillate in and out along the vertical axis an additional 5mm at every point on the trajectory. This probing behavior ensured that the tool always sucked up the blood and neither drifted above the blood nor penetrated and dragged tissue. The Robot Operation System (ROS) is used to encapsulate the image processing and robot trajectory tracking processes [30].

Roughly 50mL of liquid is injected using a syringe into the cavity at one of the four locations highlighted in Fig. 7. Before each trial, the end-effector is centered in the middle of the silicon mold as shown in Fig. 7. The percentage of liquid removed from the cavity, time to react to the injected fluid, the time to complete the trajectory were measured to evaluate the performance of the proposed automation method. The percentage of liquid removed was measured by weighing the silicon mold and syringe before and after each trial. Time to react refers to the time taken to detect the flowing blood and generate a trajectory (i.e., completing Algorithm 1) from the first moment that the injected blood is visible in the camera frame. To ensure consistency of the proposed automation method, the experiment is repeated ten times at each of the four injection spots.

The results from the total 40 trials of the automated suction experiment are shown in Table I and an example sequence is shown in Fig. 8. During the experiments, we noticed the liquid generally pooled near the bottom left injection point since it was slightly lower with respect to gravity than the rest of the cavity. This lead to shorter trajectories being generated, and hence less time to execute them as seen in the results, for the bottom left corner experiment compared to the others.

A similar set up is repeated for demonstration purposes and the result is shown in Fig. 9. The mold used for this demonstration was constructed using candle wax and has pig intestine embedded in it. Everything else is kept the same as the previously described. Despite changes to the visual textures and topology of the scene, we can see that the method is robust enough to perform autonomous tracking and suction of blood without modification.

| Injection Point | Liquid Suctioned | Time to React | Time to Execute |
|-----------------|-----------------|---------------|-----------------|
| Top Left        | 96.7%           | 4.1s          | 45.3s           |
| Top Right       | 93.6%           | 2.6s          | 47.4s           |
| Bottom Left     | 96.1%           | 4.7s          | 23.9s           |
| Bottom Right    | 96.8%           | 6.4s          | 38.1s           |

Fig. 8: Sequence of figures (from left to right) of an automated suction experiment where the liquid is injected at the bottom left corner. The images are captured from the endoscopic view with the first one highlighting the detected and tracked blood region in white and the generated trajectory in blue. To achieve real-time capabilities and react quickly to flowing blood, the blood detection and tracking algorithm is set to a lower resolution in these experiments. Nonetheless, the robotic suction tool follows the generated trajectory and efficiently removes fluids from the cavity.

TABLE 1: Mean results from automated suction experiment at each injection point.
at any time. To ensure robustness against blood stains, the algorithm relies on temporal information for detection. The novel blood flow detection and tracking algorithm presented offers a unique, probabilistic solution to tracking liquids over 3D cavities and channels, under noisy and harsh visibility conditions, and is a critical perceptual element. This estimation and tracking helps inform a trajectory generation technique to act upon the detected blood is also presented and uses a clearance reward to maximize the blood suctioned by the suction tool and be robust against imperfect blood region estimation.

For future work, we intend to push towards a complete solution for automation of hemostasis. To accomplish this, a more precise estimation of the source is needed which will be investigated by using a particle based motion model, similar to particle based physics simulators, in the temporal filtering. This should improve the blood flow tracking performance since it will have a more exact prediction step. In addition, the motion model can also be used as a forward prediction for model predictive control techniques so the suction tool can anticipate, rather than reacting, to the blood flow. Fine tissue manipulation policies that have not been explored, such as vessel grasping and clipping, will be investigated, likely leveraging open-sourced platforms like the da Vinci Reinforcement Learning (dVRL) toolkit [3].

VI. ACKNOWLEDGEMENTS

F. Richter is supported by the NSF Graduate Research Fellowship. The authors would like to thank Intuitive Surgical Inc. for instrument donations, Jingpei Lu for his support with the molds, Simon DiMiao, Omid Maharer, Dale Bergman, and Anton Deugut for their support with the dVRK.

REFERENCES

[1] M. Yip and N. Das, “Robot autonomy for surgery,” in Encyclopedia of Medical Robotics, ch. 10, pp. 281–313, World Scientific, 2017.
[2] P. Kazanzides, Z. Chen, A. Deugut, G. S. Fischer, R. H. Taylor, and S. P. DiMaio, “An open-source research kit for the da vinci® surgical system,” in 2014 IEEE International Conference on Robotics and Automation (ICRA), pp. 6434–6439, IEEE, 2014.
[3] F. Richter, R. K. Orosco, and M. C. Yip, “Open-sourced reinforcement learning environments for surgical robotics,” arXiv preprint arXiv:1903.02090, 2019.
[4] A. Murali, S. Sen, B. Kehoe, A. Garg, S. McFarland, S. Patil, W. D. Boyd, S. Lim, P. Abbeel, and K. Goldberg, “Learning by observation for surgical subtasks: Multilateral cutting of 3d viscoelastic and 2d orthotropic tissue phantoms,” in 2015 IEEE International Conference on Robotics and Automation (ICRA), pp. 1202–1209, IEEE, 2015.
[5] B. Thananjeyan, A. Garg, S. Krishnan, C. Chen, L. Miller, and K. Goldberg, “Multilateral surgical pattern cutting in 2d orthotropic gauze with deep reinforcement learning policies for tensioning,” in 2017 IEEE International Conference on Robotics and Automation (ICRA), pp. 2371–2378, IEEE, 2017.
[6] C. D’Ettorre, G. Dwyer, X. Du, F. Chadebecq, F. Vascconcelos, E. De Momi, and D. Stoyanov, “Automated pick-up of suturing needles for robotic surgical assistance,” in 2018 IEEE International Conference on Robotics and Automation (ICRA), pp. 1370–1377, IEEE, 2018.
[7] S. Sen, A. Garg, D. V. Gealy, S. McKinley, Y. Jen, and K. Goldberg, “Automating multi-throw multilateral surgical suturing with a mechanical needle guide and sequential convex optimization,” in 2016 IEEE International Conference on Robotics and Automation (ICRA), pp. 4178–4185, IEEE, 2016.
[8] F. Zhong, Y. Wang, Z. Wang, and Y.-H. Liu, “Dual-arm robotic needle insertion with active tissue deformation for autonomous suturing,” IEEE Robotics and Automation Letters, vol. 4, no. 3, pp. 2669–2676, 2019.
[9] B. Kehoe, G. Kahn, J. Mahler, J. Kim, A. Lee, A. Lee, K. Nakagawa, S. Patil, W. D. Boyd, P. Abbeel, et al., “Autonomous multilateral debridement with the raven surgical robot,” in 2014 IEEE International Conference on Robotics and Automation (ICRA), pp. 1432–1439, IEEE, 2014.
[10] Y. Li, F. Richter, J. Lu, E. K. Funk, R. K. Orosco, J. Zhu, and M. C. Yip, “Super: A surgical perception framework for endoscopic tissue manipulation with surgical robotics,” IEEE Robotics and Automation Letters, vol. 5, no. 2, pp. 2294–2301, 2020.
[11] J. Lu, A. Jayakumari, F. Richter, Y. Li, and M. C. Yip, “Super deep: A surgical perception framework for robotic tissue manipulation using deep learning for feature extraction,” arXiv preprint arXiv:2003.03477, 2020.
[12] M. Liedlgruber and A. Uhl, “Computer-aided decision support systems for endoscopy in the gastrointestinal tract: a review,” IEEE reviews in biomedical engineering, vol. 4, pp. 73–88, 2011.
[13] Y. Fu, W. Zhang, M. Mandal, and M. Q.-H. Meng, “Computer-aided bleeding detection in wce video,” IEEE journal of biomedical and health informatics, vol. 18, no. 2, pp. 636–642, 2013.
[14] J. Liu and X. Yuan, “Occlusion bleeding detection in endoscopy images using support vector machines,” Optimization and engineering, vol. 10, no. 2, pp. 289–299, 2009.
[15] Y. S. Jung, Y. H. Kim, D. H. Lee, and J. H. Kim, “Active blood detection in a high resolution capsule endoscopy using color spectrum transformation,” in 2008 International Conference on Biomedical Engineering and Informatics, vol. 1, pp. 859–862, IEEE, 2008.
[16] T. Okamoto, T. Ohnishi, H. Kawahira, O. Derghachyava, P. Jannin, and H. Haneishi, “Real-time identification of blood regions for hemostasis support in laparoscopic surgery,” Signal, Image and Video Processing, vol. 13, no. 2, pp. 405–412, 2019.
[17] B. Li and M. Q.-H. Meng, “Computer-aided detection of bleeding regions for capsule endoscopy images,” IEEE Transactions on biomedical engineering, vol. 56, no. 4, pp. 1032–1039, 2009.
[18] G. Pan, G. Yan, X. Qiu, and J. Cui, “Bleeding detection in wireless capsule endoscopy based on probabilistic neural network,” Journal of medical systems, vol. 35, no. 6, pp. 1477–1484, 2011.
[19] Y. Fu, M. Mandal, and G. Guo, “Bleeding region detection in wce images based on color features and neural network,” in 2011 IEEE 54th International Midwest Symposium on Circuits and Systems (MWSCAS), pp. 1–4, IEEE, 2011.
[20] R. Mottaghi, C. Schenkck, D. Fox, and A. Farhadi, “See the glass half full: Reasoning about liquid containers, their volume and content,” in Proceedings of the IEEE International Conference on Computer Vision, pp. 1871–1880, 2017.
[21] C. Schenkck and D. Fox, “Perceiving and reasoning about liquids using fully convolutional networks,” The International Journal of Robotics Research, vol. 37, no. 4-5, pp. 452–471, 2018.
[22] C. Schenkck and D. Fox, “Spnets: Differentiable fluid dynamics for deep neural networks,” arXiv preprint arXiv:1806.06094, 2018.
[23] A. Yamaguchi and C. G. Atkeson, “Stereo vision of liquid and particle flow for robot pouring,” in 2016 IEEE-RAS 16th International Conference on Humanoid Robots (Humanoids), pp. 1173–1180, IEEE, 2016.
[24] M. Allan, A. Shvets, T. Kurmann, Z. Zhang, R. Duggal, Y.-H. Su, N. Rieke, I. Laina, N. Kalavakonda, S. Bodenstedt, et al., “2017 robotic instrument segmentation challenge,” arXiv preprint arXiv:1902.06426, 2019.
[25] D. Teney and M. Hebert, “Learning to extract motion from videos in convolutional neural networks,” in Asian Conference on Computer Vision, pp. 412–428, Springer, 2016.
[26] B. D. Lucas, T. Kanade, et al., “An iterative image registration technique with an application to stereo vision,” 1981.
[27] S. Baker, D. Scharstein, J. Lewis, S. Roth, M. J. Black, and R. Szeliski, “A database and evaluation methodology for optical flow,” 1981.
[28] M. Allan, A. Shvets, T. Kurmann, Z. Zhang, R. Duggal, Y.-H. Su, N. Rieke, I. Laina, N. Kalavakonda, S. Bodenstedt, et al., “2017 robotic instrument segmentation challenge,” arXiv preprint arXiv:1902.06426, 2019.
[29] D. Teney and M. Hebert, “Learning to extract motion from videos in convolutional neural networks,” in Asian Conference on Computer Vision, pp. 412–428, Springer, 2016.
[30] B. D. Lucas, T. Kanade, et al., “An iterative image registration technique with an application to stereo vision,” 1981.
[31] S. Baker, D. Scharstein, J. Lewis, S. Roth, M. J. Black, and R. Szeliski, “A database and evaluation methodology for optical flow,” International journal of computer vision, vol. 92, no. 1, pp. 1–31, 2011.
[32] G. Farnebäck, “Two-frame motion estimation based on polynomial expansion,” in Scandinavian conference on Image analysis, pp. 363–370, Springer, 2003.
[33] J.-R. Chang and Y.-S. Chen, “Pyramidal stereo matching network,” in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 5410–5418, 2018.
[34] Stanford Artificial Intelligence Laboratory et al., “Robotic operating system.”