Autonomous Robotic Suction to Clear the Surgical Field for Hemostasis Using Image-Based Blood Flow Detection

Florian Richter, *Graduate Student Member, IEEE*, Shihao Shen, *Student Member, IEEE*, Fei Liu, Jingbin Huang, *Graduate Student Member, IEEE*, Emily K. Funk, Ryan K. Orosco, *Member, IEEE*, and Michael C. Yip, *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.**

***Index Terms*—Medical robots and systems, computer vision for medical robotics, surgical robotics, laparoscopy.**

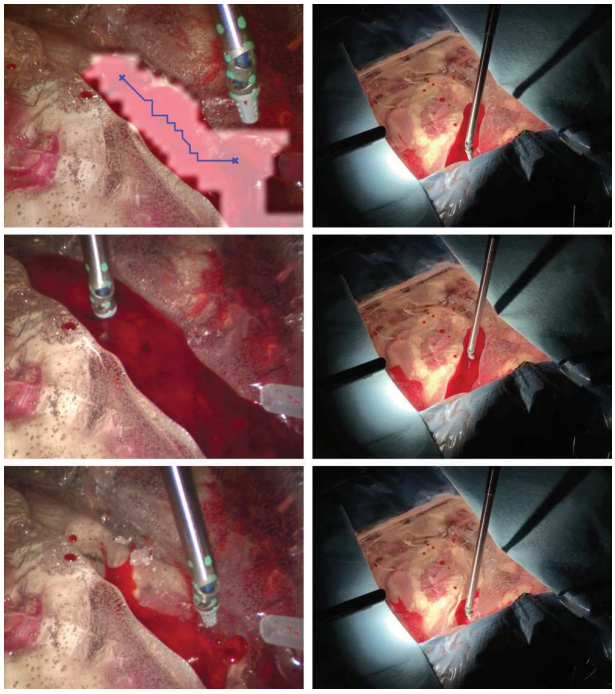

Fig. 1. Autonomous blood suction for cases of trauma in surgery. A robotic suction tool clears the surgical field of flowing blood. The pooling and direction of flowing blood is detected using a temporal, image-based algorithm. This informs a robot trajectory for the suction tool to move upstream towards the source while constrained to stay along the flowing blood.

I. Introduction

SINCE the deployment of surgical robotic devices such as the da Vinci 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 surgeries itself to address lack-of-access to timely, life-saving surgery in remote or under-served communities. 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

Manuscript received October 15, 2020; accepted January 17, 2021. Date of publication February 1, 2021; date of current version February 23, 2021. This letter was recommended for publication by Associate Editor E. De Momi and Editor P. Valdastri upon evaluation of the reviewers' comments. This work was supported in part by the National Science Foundation (NSF) under Grants 1830403 and 1935329 and in part by the US Army Telemedicine and Advanced Technology Research Center under the Robotic Battlefield Medical Support System project. The work of F. Richter was supported by the NSF Graduate Research Fellowship. *(Corresponding author: Florian Richter.)*

Florian Richter, Shihao Shen, Fei Liu, Jingbin Huang, and Michael C. Yip are with the Department of Electrical and Computer Engineering, University of California San Diego, La Jolla, CA 92093 USA (e-mail: frichter1995@gmail.com; s1shen@ucsd.edu; f4liu@ucsd.edu; jih023@ucsd.edu; yip@ucsd.edu).

Emily K. Funk and Ryan K. Orosco are with the Department of Surgery - Division of Head and Neck Surgery, University of California San Diego, La Jolla, CA 92093 USA (e-mail: ekfunk@ucsd.edu; ryanorosco@gmail.com).

This letter has supplementary downloadable material available at https://doi.org/10.1109/LRA.2021.3056057, provided by the authors.

Digital Object Identifier 10.1109/LRA.2021.3056057

2377-3766 © 2021 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission.
See https://www.ieee.org/publications/rights/index.html for more information.

Authorized licensed use limited to: Univ of Calif San Diego. Downloaded on August 31,2021 at 04:55:49 UTC from IEEE Xplore. Restrictions apply.

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. If surgical automation is ever to be deployed, reactive policies will be required to handle these traumatic scenarios.

This work specifically addresses the problem of small and medium vessel ruptures. 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 letter, 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,
3) and 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 a 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.

Algorithm 1: Blood Suction.

Initialize image frame probability map of blood: $\mathbf{P}_0$
Initialize optical flow CNN with l image frame inputs
Initialize counts of being blood: $\mathbf{C}_t \leftarrow \mathbf{0}$
while *new image,* $\mathbf{I}_t$*, arrives* **do**
  // Blood Region Detection
  $\mathbf{O}_t \leftarrow$ getOpticalFlowCNN$(\mathbf{I}_t, \ldots, \mathbf{I}_{t-l-1})$
  for *pixel* $p_t \in \mathbf{P}_t$ **do**
    // Detect:
    if $||\mathbf{O}_t(p)|| > \gamma_o$ **then**
      $z_t^p \leftarrow$ b
    else
      $z_t^p \leftarrow \overline{\text{b}}$
    Predict: $P(p_t = \text{b}|z_{1:t-1}^p)$ with (8) and $\mathbf{P}_{t-1}$
    Update: $P(p_t = \text{b}|z_{1:t}^p)$ with (7) and z_t^p
  $\mathbf{B}_t$ is set to the largest connected region in binary mask $\mathbf{P}_t > 0.5$
  if $size(\mathbf{B}_t) < \gamma_B$ **then**
    continue
  // Suction Trajectory Generation
  Update Blood Count: $\mathbf{C}_t \leftarrow \mathbf{C}_{t-1} + \mathbf{B}_t$
  End point: $e_t \leftarrow \underset{pixel}{\text{argmax}}\ \mathbf{C}_t \wedge \mathbf{B}_t$
  Start point: $s_t \leftarrow \underset{pixel}{\text{argmin}}\ \mathbf{C}_t \wedge \mathbf{B}_t$
  $\mathbf{T}_t \leftarrow$ generateTrajectoryForSuction$(s_t, e_t, \mathbf{B}_t)$
  if *length*$(\mathbf{T}_t) > \gamma_T$ **then**
    break
executeTrajectory$(\mathbf{T}_t)$

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.

Authorized licensed use limited to: Univ of Calif San Diego. Downloaded on August 31,2021 at 04:55:49 UTC from IEEE Xplore. Restrictions apply.

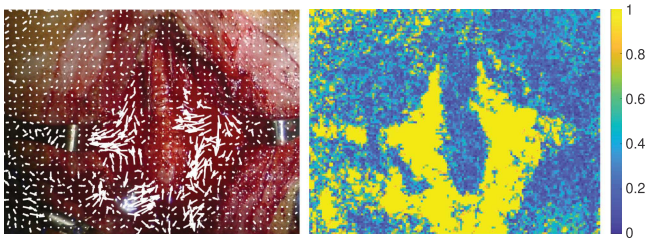

Fig. 2. The above images are generated by optical flow estimation from an in-vivo surgical scene. The left image shows the vectors of estimated image motion from optical flow, and the right image is a normalized heatmap of their magnitudes. Notice that the magnitude of optical flow detects the regions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

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. 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.

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, $z_t^p = \mathrm{b}$, if the magnitude of optical flow at pixel p_t is greater than a threshold, γ_o. The inverse is also set, so no blood is detected, $z_t^p = \bar{\mathrm{b}}$, if the magnitude at pixel p_t is less than γ_o. Hence the probability model for these detections can be simply written as:

$$P(z_t^p = \mathrm{b} \mid p_t = \mathrm{b}) \qquad P(z_t^p = \mathrm{b} \mid p_t = \bar{\mathrm{b}}) \tag{1}$$

which describes an observation model for the hidden state $p_t \in \{\mathrm{b}, \bar{\mathrm{b}}\}$.

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

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

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:

$$P(p_{t+1} = \mathrm{b} \mid p_t = \bar{\mathrm{b}}, k_t^p = \mathrm{b}) \tag{3}$$

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

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

$$k_t^p = \bigvee_{q^i \in \mathbf{A}_p} q_t^i \tag{5}$$

where $\mathbf{A}_p$ 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. To appropriately describe these processes in this case of blood vessel ruptures, (3) should be set less than (2) and (4) close to 0.

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:

$$\begin{aligned} P(p_{t+1} \mid z_{1:t}^p) &= P(p_{t+1} \mid p_t = \mathrm{b}) P(p_t = \mathrm{b} \mid z_{1:t}^p) \\ &+ \sum_{k_t^p \in \{\mathrm{b}, \bar{\mathrm{b}}\}} P(p_{t+1} \mid p_t = \bar{\mathrm{b}}, k_t^p) P(p_t = \bar{\mathrm{b}}, k_t^p \mid z_{1:t}^p) \end{aligned} \tag{6}$$

and the update step is computed:

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

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

Authorized licensed use limited to: Univ of Calif San Diego. Downloaded on August 31,2021 at 04:55:49 UTC from IEEE Xplore. Restrictions apply.

Algorithm 2: Trajectory Generation for Suction.

Input : Start pixel, s_t, end pixel, e_t, and mask of blood region $\mathbf{B}_t$
Output: Trajectory $\mathbf{T}_t$

Initialize clearance reward $\mathbf{R} \leftarrow \mathbf{0}$
Initialize temporary eroded blood region $\mathbf{E} \leftarrow \mathbf{B}_t$
Initialize $i \leftarrow 0$
while $\mathbf{E} \neq \mathbf{0}$ *and* $i < \gamma_r$ **do**
  $\mathbf{E} \leftarrow \text{erode}(\mathbf{E})$
  $\mathbf{R} \leftarrow \mathbf{R} + r\mathbf{E}$
  $i \leftarrow i + 1$
Initialize pixels cost to go to goal: $\mathbf{D} \leftarrow \infty$
Initialize visited map: $\mathbf{V} \leftarrow !\mathbf{B}_t$
Initialize parents of nodes: $\mathbf{K} \leftarrow$ UNDEFINED
$\mathbf{D}(s_t) \leftarrow 0$
while $\mathbf{V} \neq \mathbf{1}$ **do**
  $u \leftarrow \underset{pixel}{\text{argmin}}\ \mathbf{D}$
  $\mathbf{V}(u) \leftarrow 1$
  if $u = e_t$ **then**
    break
  for *pixel* q *neighboring* u *and* $\mathbf{V}(q) \neq 1$ **do**
    if $\mathbf{D}(q) > ||q - u|| - \mathbf{R}(q)$ **then**
      $\mathbf{D}(q) \leftarrow ||q - u|| - \mathbf{R}(q)$
      $\mathbf{K}(q) \leftarrow u$
Initialize trajectory for output: $\mathbf{T}_t \leftarrow [\ ]$
Initialize parent traversal node: $u \leftarrow e_t$
while $u \neq s_t$ **do**
  Insert u at beginning of $\mathbf{T}_t$
  $u \leftarrow \mathbf{K}(p)$
return $\mathbf{T}_t$

blood is approximated to be independent of all others at time t. With this simplification, the predict step can be rewritten as:

$$P(p_{t+1} \mid z^p_{1:t}) = P(p_{t+1} \mid p_t = \mathrm{b})P(p_t = \mathrm{b} \mid z_{1:t}) + \sum_{k^p_t \in \{\mathrm{b},\bar{\mathrm{b}}\}} P(p_{t+1} \mid p_t = \bar{\mathrm{b}}, k^p_t)P(k^p_t \mid z^p_{1:t})P(p_t = \bar{\mathrm{b}} \mid z^p_{1:t}) \tag{8}$$

and an expression can be found for $P(k^p_t \mid z^p_{1:t})$ using the inclusion-exclusion principle:

$$P(k^p_t \mid z^p_{1:t}) = \sum_{j=1}^{|\mathbf{A}_p|} (-1)^{j-1} \sum_{\substack{\mathbf{J} \subseteq \mathbf{A}_p \\ |\mathbf{J}| = j}} \prod_{q^i \in \mathbf{J}} P(q^i_t \mid z^p_{1:t}) \tag{9}$$

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 γ_B. This threshold keeps a detection from occurring when there is no actual blood flowing.

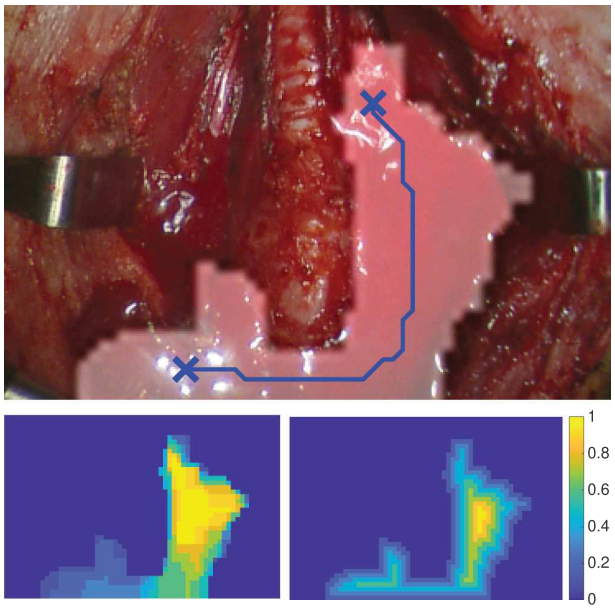

Fig. 3. Example of trajectory generation for blood suction from in-vivo scene. In the top figure, the white highlighted region is the detected blood region and the blue path is the generated trajectory. The starting and stopping points are chosen from the newest and oldest regions of the detected blood 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.

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 γ_r iterations. The pixels left in the eroded region are given an additional reward of r at each iteration. The final trajectory in the image frame is then generated using Dijkstra's algorithm where the path is constrained to stay within the blood region and the clearance reward is subtracted from the normal distance cost. An outline of this trajectory generation technique is shown in Algorithm 2, and an example is shown in Fig. 3. The trajectory is then executed if it is longer than a threshold γ_T. This threshold gives time for

Authorized licensed use limited to: Univ of Calif San Diego. Downloaded on August 31,2021 at 04:55:49 UTC from IEEE Xplore. Restrictions apply.

Fig. 4. Simulated scenes used to evaluate the proposed blood flow detection algorithm. The arrows highlight the direction of flow for the blood and how it fills the cavity. The detection algorithm estimates the blood flow via temporal tracking and aids in trajectory generation for a suction tool to remove the blood.

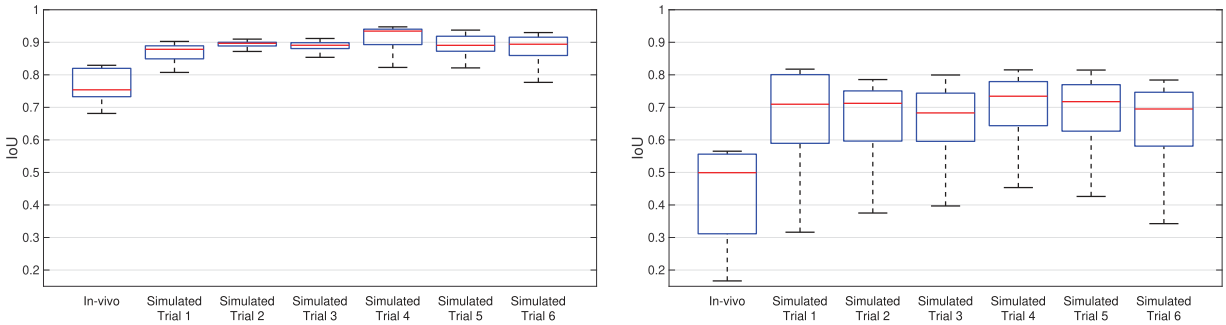

Fig. 5. The left and right plots show Intersection over Union (IoU) results of blood flow estimation when using the proposed blood flow detection and tracking algorithm and only the blood detection from optical flow thresholding respectively. The complete pipeline results in better performance overall.

the start and end points to stabilize so an effective trajectory can be generated.

IV. Experiments and Results

The proposed blood flow detection and tracking method was evaluated on both simulated scenes and a live surgery involving a hemorrhage during a thyroidectomy. The complete automated suctioning solution was demonstrated on in a lab setting on a simulated surgical cavity and red fluid for blood. The following sections describe these experiments, the necessary implementation details, and results.

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], uses $l = 3$ image frames for input, and the resolution of the optical flow estimation is 1/4 of the input frame resolution. These are the default values of the original CNN implementation. The size of the probability map is set to the optical flow resolution. The threshold for detection, γ_o, region size, γ_B, maximum number of erosions for clearance, γ_r, and trajectory length, γ_T, are set to 0.45, 20, 4, and 30 respectively. The detection probability, $P(z_t^p = \mathrm{b}|p_t = \mathrm{b})$, $P(z_t^p = \mathrm{b}|p_t = \overline{\mathrm{b}})$, are set to 0.95 and 0.2 respectively because experimentally we found the true positive rate and false negative rate to be very accurate and noisy respectively. The initial probability of a pixel being blood, $P(p_0 = \mathrm{b})$ and transition probabilities of a pixel being blood, $P(p_{t+1} = \mathrm{b}|p_t = \mathrm{b})$, $P(p_{t+1} = \mathrm{b}|p_t = \overline{\mathrm{b}}, k_t^p = \mathrm{b})$, $P(p_{t+1} = \mathrm{b}|p_t = \overline{\mathrm{b}}, k_t^p = \overline{\mathrm{b}})$, are set to 0.1, 0.98, 0.85, and 0.01 respectively. The neighbors for a pixel, $\mathbf{A}_p$, are set to just up, down, left, and right since the algorithm needs to run quickly for real-time detection in the upcoming experiments. The clearance reward per erosion, r, is set to 0.2.

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, $\mathbf{G}_t$, of the blood region. Performance is evaluated from these datasets using the Intersection over Union (IoU) metric:

$$\frac{\mathbf{B}_t \wedge \mathbf{G}_t}{\mathbf{B}_t \vee \mathbf{G}_t} \tag{10}$$

where $\mathbf{B}_t$ is a mask of the detected blood region from our proposed method, $\wedge$ is the Boolean-AND operation and $\vee$ 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, $\mathbf{G}_t$, 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.

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 8 s 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, $\mathbf{G}_t$, 10 evenly distributed frames were manually annotated. The recorded image size was 640 by 480 pixels.

Authorized licensed use limited to: Univ of Calif San Diego. Downloaded on August 31,2021 at 04:55:49 UTC from IEEE Xplore. Restrictions apply.

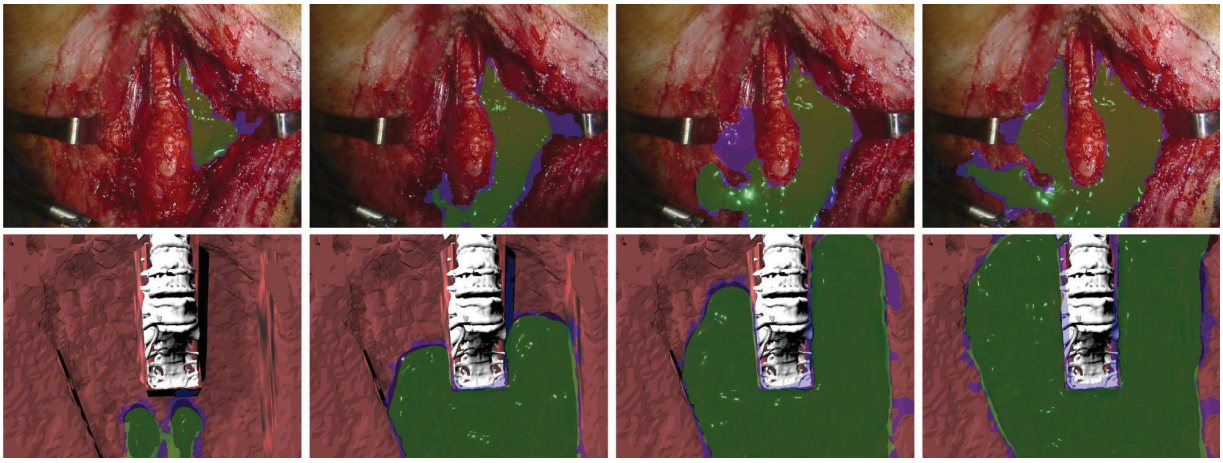

Fig. 6. Example figures of blood flow detection and tracking. The top and bottom sequences are from an in-vivo rupture and simulated scene respectively. The highlighted green regions are the intersection between ground truth and blood flow detection. Meanwhile, the highlighted blue regions are the error of detection, i.e., the union minus intersection between the ground truth and blood flow detection. These figures are best viewed in color.

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 γ_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.

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 30 fps 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 640 by 480 pixels meanwhile the blood flow detection algorithm used a reduced image size of 160 by 120 pixels to improve its speed.

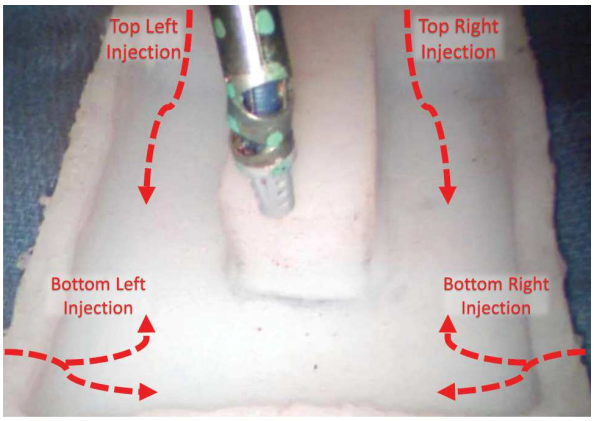

Fig. 7. Endoscopic view of phantom used for the automated suction experiments. The red arrows highlight the direction of flow at the four injection points tested in the experiment.

A Patient Side Manipulator (PSM) from the dVRK [2] was equipped with 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, $\mathbf{b}_t$, is iteratively set to:

$$\overline{\mathbf{b}}_{t+1} = \begin{cases} \gamma_s \frac{\mathbf{d}_t}{||\mathbf{d}_t||} + \overline{\mathbf{b}}_t & \text{if } ||\mathbf{d}_t|| > \gamma_s \\ \mathbf{d}_t + \overline{\mathbf{b}}_t & \text{if } ||\mathbf{d}_t|| \leq \gamma_s \end{cases} \tag{11}$$

where $\gamma_s = 0.75$ mm is the max step size, the operator $\bar{\cdot} = [\cdot \;\; 1]^\top$ gives the homogeneous representation of a point, and the direction, $\mathbf{d}_t$, is computed as

$$\mathbf{d}_t = \mathbf{T}_c^b \overline{\mathbf{b}}^g - \overline{\mathbf{b}}_t \tag{12}$$

The camera to base transform, $\mathbf{T}_c^b \in SE(3)$, is estimated in real time using our previous work [10] and $\mathbf{b}^g$ is the 3D goal position

Authorized licensed use limited to: Univ of Calif San Diego. Downloaded on August 31,2021 at 04:55:49 UTC from IEEE Xplore. Restrictions apply.

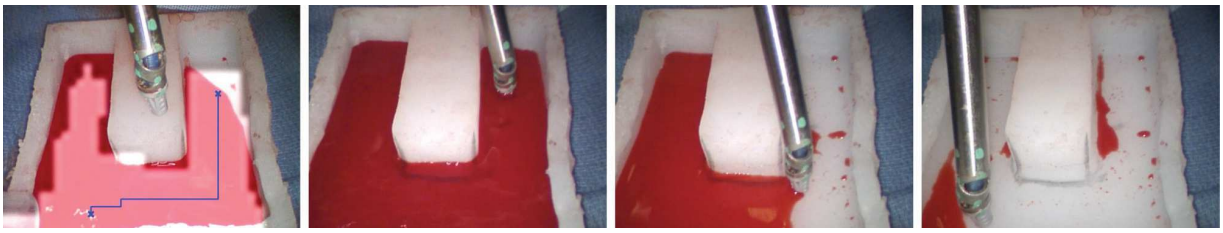

Fig. 8. Sequence of figures (from left to right) of an automated suction experiment where the liquid is injected at the bottom left corner. The images are captured from the endoscopic view with the first one highlighting the detected and tracked blood region in white and the generated trajectory in blue. 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 I
MEAN RESULTS FROM AUTOMATED SUCTION EXPERIMENT AT EACH INJECTION POINT

Injection Point	Liquid Suctioned	Time to React	Time to Execute Trajectory
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

generated by the trajectory and PSMNet. This controller is ran at 100 Hz and is repeated until $||\mathbf{d}_t|| < 2$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, $\mathbf{b}_t$, 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 direction of gravity an additional 5 mm 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 50 mL 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 such that it does not obstruct any of the injected liquid 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. 1. 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.

V. DISCUSSION AND CONCLUSION

In this work, the first completely automated solution for clearing the surgical field from blood is presented. The solution provides both the perception, trajectory generation, and control strategy required for the task of clearing blood. This is the first step taken towards automation of a crucial surgical task, hemostasis, which can occur in any surgery 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 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 location of the bleeding source will be estimated using a particle based motion model, similar to PBD simulators, in the temporal filtering. Another consideration will be integration of surgical tool masking using our previous works [10], [11] into the blood tracking framework as the hemostasis automation task will require additional surgical robotic tools.

ACKNOWLEDGMENT

The authors would like to thank Intuitive Surgical Inc. for instrument donations, Jingpei Lu for his support with PSMNet, Harleen Singh for her support with the molds, Simon DiMiao, Omid Maherari, Dale Bergman, and Anton Deguet for their support with the dVRK.

Authorized licensed use limited to: Univ of Calif San Diego. Downloaded on August 31,2021 at 04:55:49 UTC from IEEE Xplore. Restrictions apply.

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. Deguet, G. S. Fischer, R. H. Taylor, and S. P. DiMaio, "An open-source research kit for the da vinci surgical system," in *Proc. IEEE Int. Conf. Robot. Auto.*, 2014, pp. 6434–6439.

[3] F. Richter, R. K. Orosco, and M. C. Yip, "Open-sourced reinforcement learning environments for surgical robotics," 2019, *arXiv:1903.02090*.

[4] A. Murali *et al.*, "Learning by observation for surgical subtasks: Multilateral cutting of 3 d viscoelastic and 2 d orthotropic tissue phantoms," in *Proc. IEEE Int. Conf. Robot. Auto.*, 2015, pp. 1202–1209 .

[5] B. Thananjeyan, A. Garg, S. Krishnan, C. Chen, L. Miller, and K. Goldberg, "Multilateral surgical pattern cutting in 2 d orthotropic gauze with deep reinforcement learning policies for tensioning," in *Proc. IEEE Int. Conf. Robot. Auto.*, 2017, pp. 2371–2378 .

[6] C. D'Ettorre *et al.*, "Automated pick-up of suturing needles for robotic surgical assistance," in *Proc. IEEE Int. Conf. Robot. Auto.*, 2018, pp. 1370–1377.

[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 *Proc. IEEE Int. Conf. Robot. Auto.*, 2016, pp. 4178–4185.

[8] F. Zhong, Y. Wang, Z. Wang, and Y.-H. Liu, "Dual-arm robotic needle insertion with active tissue deformation for autonomous suturing," *IEEE Robot. Automat. Lett.*, vol. 4, no. 3, pp. 2669–2676, Jul. 2019.

[9] B. Kehoe *et al.*, "Autonomous multilateral debridement with the raven surgical robot," in *Proc. IEEE Int. Conf. Robot. Auto.*, 2014, pp. 1432–1439.

[10] Y. Li *et al.*, "Super: A surgical perception framework for endoscopic tissue manipulation with surgical robotics," *IEEE Robot. Automat. Lett.*, vol. 5, no. 2, pp. 2294–2301, Apr. 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," 2020, *arXiv:2003.03472*,.

[12] M. Liedlgruber and A. Uhl, "Computer-aided decision support systems for endoscopy in the gastrointestinal tract: A review," *IEEE Revi. Biomed. Eng.*, 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 J. Biomed. Health Infor.*, vol. 18, no. 2, pp. 636–642, Mar. 2014.

[14] J. Liu and X. Yuan, "Obscure bleeding detection in endoscopy images using support vector machines," *Optim. Eng.*, 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 *Proc. Int. Conf. BioMed. Eng. Informat.*, 2008, pp. 859–862.

[16] T. Okamoto, T. Ohnishi, H. Kawahira, O. Dergachyava, P. Jannin, and H. Haneishi, "Real-time identification of blood regions for hemostasis support in laparoscopic surgery," *Signal, Image Video Process.*, 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 Trans. Biomed. Eng.*, vol. 56, no. 4, pp. 1032–1039, Apr. 2009.

[18] G. Pan, G. Yan, X. Qiu, and J. Cui, "Bleeding detection in wireless capsule endoscopy based on probabilistic neural network," *J. Med. Syst.*, 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 *Proc. IEEE 54th Int. Midwest Symp. Circuits Syst.*, 2011. pp. 1–4 .

[20] R. Mottaghi, C. Schenck, D. Fox, and A. Farhadi, "See the glass half full: Reasoning about liquid containers, their volume and content," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2017, pp. 1871–1880.

[21] C. Schenck and D. Fox, "Perceiving and reasoning about liquids using fully convolutional networks," *The Int. J. Robot. Res.*, vol. 37, no. 4-5, pp. 452–471, 2018.

[22] C. Schenck and D. Fox, "Spnets: Differentiable fluid dynamics for deep neural networks," *Conf. Robot Learn.*, 2018, pp. 317–335.

[23] A. Yamaguchi and C. G. Atkeson, "Stereo vision of liquid and particle flow for robot pouring," in *Proc. IEEE-RAS 16th Int. Conf. Humanoid Robot.*, 2016, pp. 1173–1180.

[24] M. Allan *et al.*, "2017 robotic instrument segmentation challenge," 2019, *arXiv:1902.06426*,.

[25] D. Teney and M. Hebert, "Learning to extract motion from videos in convolutional neural networks," in *Proc. Asian Conf. Comput. Vis.*, 2016, pp. 412–428 .

[26] B. D. Lucas, T. Kanade, "An Iterative Image Registration Technique With an Application to Stereo Vision," in *Proc. 7th Int. Joint Conf. Artif. Intell.*, 1981, pp. 674-679.

[27] S. Baker, D. Scharstein, J. Lewis, S. Roth, M. J. Black, and R. Szeliski, "A database and evaluation methodology for optical flow," *Int. J. Comput. Vis.*, vol. 92, no. 1, pp. 1–31, 2011.

[28] G. Farnebäck, "Two-frame motion estimation based on polynomial expansion," in *Proc. Scandinavian Conf. Image Anal.*, 2003, pp. 363–370.

[29] J.-R. Chang and Y.-S. Chen, "Pyramid stereo matching network," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 5410–5418.

[30] M. Quigley *et al.*, "ROS: an open-source Robot Operating System," *ICRA Workshop Open Source Software*, vol. 3, no. 3.2 p. 5, 2009.

Authorized licensed use limited to: Univ of Calif San Diego. Downloaded on August 31,2021 at 04:55:49 UTC from IEEE Xplore. Restrictions apply.