Web Based Camera Navigation for Virtual Pancreatic cancer surgery: Whipple Surgery Simulator (VPanSS)

Doga Demirel, Alexander Yu, Tansel Halic, Sinan Kockara

Computer Science Department
University of Central Arkansas

***Abstract*— This study presents an preliminary design and development of a web based real-time virtual laparoscopy simulation for pancreticoduodentomy (Whipple Procedure). In this early stage, we primarily focused on virtual camera navigation to improve hand-eye coordination of a surgeon. Pancreticoduodentomy, pancreatic surgery, is one of most challenging laparoscopic surgeries. We examined whole Whipple procedure to identify its major tasks for the design of a virtual training simulator. Based on the tasks analysis, we found that one of the major challenges in Whipple surgery is accessing to the pancreas. Therefore, identification of the tumorous regions and tumor spread around pancreas become complicated. Surgeons need to carefully navigate laparoscopic camera to the pancreas and nearby tissues and locate the regions affected by the tumor. However, cameras navigation in laparoscopy is intricate due to different abdominal entry locations of camera and surgical tools. Therefore, we developed a real-time, web based, surgical simulation training platform for camera navigation task for Whipple procedure. The goal is to provide accessible, portable, ubiquitous, hardware independent training simulation platform unlike any other surgical simulators. In VPanSS, we developed and integrated a novel contact detection algorithm to continuously determine the camera-organ/tissue contacts during the laparoscopic camera navigation. We tested real-time performance of VPanSS on various platforms to understand its effectiveness and applicability.**

***Index Terms*— Computer graphics, Medical simulation, Internet, Cancer, Tumors**

I. INTRODUCTION

This paper describes the preliminary work on the design and development of virtual reality based surgery simulation for pancreatic cancer removal. Having one of the lowest survival rates among all cancers, pancreatic cancer is one of the deadliest types. Pancreatic cancer surgery has a low survival rate [1]. It is one of the most complex surgeries to carry out successfully. The pancreas is difficult to reach because of its location that is in the abdomen and posterior to the stomach, anterior to the spine. Anatomical location of pancreas makes any surgical treatment difficult. Its location also makes challenging to detect the tumor growth with basic medical examination. The pancreatic cancer does not cause any symptoms until the tumor grows large. Pancreatic cancer can also grow into the nearby organs and tissues making it even more deadly. This increases the complexity of any surgical intervention of the tumor removal. It is shown that when pancreatic operation is performed by surgeons with minimal experience, it has three times higher mortality rate than operation applied by experienced surgeons [2]. There is an apparent need for a risk-free training platform to increase surgeons' experiences.

Only 10% to 20% of the people that are diagnosed with pancreatic cancer are suitable for a surgery [3]. Before a surgeon makes a decision about the patient's suitability for the surgery, s/he will examine the size of the tumor, location in the pancreas and whether the cancer has grown into adjacent tissues, lymph nodes, blood vessels or any other part of the body. Although advancements in current imaging techniques improve the assessment of the tumor's spread, they are often not adequate. During the procedure, the surgeon's evaluation and exact identification of the regions is needed. The surgery necessitates dynamic decision making which is specific to the patient [4]. This increases the level of complexity of the surgery.

There are 3 common curative procedures performed for the removal of pancreatic cancer. The first curative procedure is called the distal pancreatectomy. It is the operation that removes various parts of the pancreas such as the tail, or in some cases the tail and a part of the body and the spleen of the pancreas. Distal pancreatectomy has only a mortality rate of 5%, and provides pain relief for 80% of the patients [3]. Another curative procedure is called the total pancreatectomy which removes the entire pancreas and the spleen [5]. Last and the most common one is called the pancreatoduodenectomy or the "Whipple." "Whipple" procedure is one of the most risky and demanding operations for surgeons and patients. In this procedure, the head of the pancreas, the gallbladder, the duodenum and in some cases the body of the pancreas are removed. After the removal of infected parts is completed, the surgeon reconnects the remaining pancreas and digestive organs. This connectivity allows pancreatic enzymes, bile, and stomach contents to flow into the small intestine during digestion.

The study revealed that patients undergo the "Whipple" procedure has better recovery rate when the surgery is performed at a hospital that performs the procedure around twenty times a year [2]. The study states that when the "Whipple" procedure is conducted in a high-volume hospital (16 or more procedures a year) with experienced surgeons, the mortality rate of surgical complications is recorded as 3.8%. On the contrary, when the surgery is performed in a low-volume (less than 16 procedures a year) hospital with less

experienced surgeons the rate quadruples to 16.3% [2]. With that percent difference, it is one of the highest among other cancers. The experience of the surgeons in hospital can also affect the length of the patient stay. Performed in a high volume hospital the length of stay is 18.2 days, in a low volume hospital the number increases to 23.6 days [5].

Pancreas surgery requires highly experienced and skilled surgeons. Considering the complexity and post-surgery complications of the surgery, the patient recovery and lifespan are highly correlated with the skills set and knowledge of the surgeons. As the statistics reveals high mortality amongst the novice surgeons, the demand in risk free training is acute. However, there is no available risk-free training platform for surgeons that allow them to enhance their skill and gain experience on sub-tasks of the procedure. The conventional trainings such as practicing on cadavers or animals are not sufficient for complex surgery like pancreticoduodentomy [6]. Moreover, performing on real patients entails high risk. Therefore, we analyzed the most critical tasks in the pancreatic surgery and developed preliminary virtual training platform. The major contributions of this study are two-fold; one is detailed task analysis of pancreticoduodentomy and second is the web based virtual camera navigation task to improve hand-eye coordination of surgeons, which is noted as one of the most critical skills for surgeons in the "Whipple" surgery. Furthermore, we presented performance tests to convey the effectiveness simulator on various platforms.

II. Task Analysis

We analyzed tasks of the "Whipple" procedure [7, 8, 9]. Every task was detailed and sub-tasks were derived. Out of the 3 curative procedures, "Whipple" surgery is the most common operation to treat pancreatic cancer. The range of the "Whipple" procedure is between 370-660 minutes [7]. In table I, the 10 major tasks are shown.

TABLE I
TEN MAJOR TASKS FOR PANCRETICODUODENTOMY

Step	Sub Task
1	Stabilizing the Stomach
2	Freeing 1st and 2nd part of Duodenum
3	Creating a Passage
4	Freeing 3rd and 4th part of Duodenum
5	Cystic Artery Removal
6	Jejunum
7	Removing the Tumor
8	Pancreaticojejunostomy
9	Hepaticojejunostomy
10	Duodenojejunostomy

Figures (Fig. 1a, 1b, 1c, 1d and 1e) show the task analysis trees for the "Whipple" procedure. Before surgery starts a laparoscopic telescope is inserted to check the surrounding organs for cancer [8]. The patient is set up for the laparoscopic surgery with 4 holes, one for camera, one for left and one for right hand working and one for retraction [9].

A. Stabilizing the Stomach and Freeing 1st and 2nd part of Duodenum

The operation starts by laparoscopic forceps grasping the greater curvature of the stomach and retracting it upward (Fig. 1a) [10]. When the stomach is out of the way for the surgery, laparoscopic scissor blades are inserted behind duodenum to free the 1st and the 2nd part of duodenum (Fig. 1a).

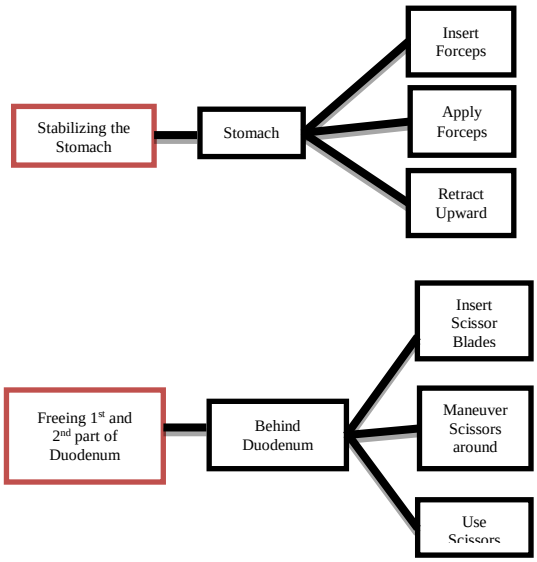

Fig. 1a Subtasks stabilizing the stomach and freeing 1st and 2nd part of duodenum in the hierarchical task analysis tree for pancreticoduodentomy

B. Creating a Passage and Freeing 3rd and 4th Parts of Duodenum

Linear stapler is a cutting tool used to remove the cancer part of the pancreas. A passage will be created to move the linear stapler to the pancreas by dividing the portion of the greater omentum that extends from the transverse colon to the greater curvature of the stomach by the linear stapler (Fig. 1b) [11]. The muscle that connects the duodenum to the diaphragm also known as the suspensory muscle of duodenum is cut with Laparoscopic scissor blades to free the parts of duodenum (Fig. 1b) [8].

C. Cystic Artery Removal and Jejunum

When the duodenum is freed, the right gastro-omental vein and the right gastric vessels are exposed; using a hook cautery these vessels are divided and then sutured (Fig. 1c) [12].The cystic artery can be located in the hepatoduodenal ligament. After finding the cystic artery, it is clipped, sutured and divided (Fig. 1c) [10]. The cystic duct is then separated from the fibrous tissue using a Laparoscopic knife. After freeing the duodenum, next step is to cut an incision across the jejunum using a linear gastrointestinal stapler (Fig. 1c) [10]. The incision is then clamped and sutured.

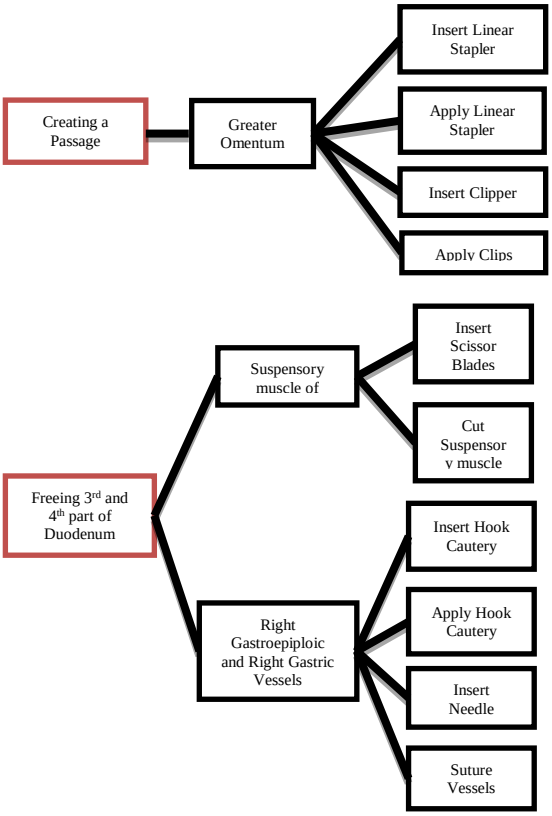

Fig. 1b Subtasks creating a passage and freeing 3rd and 4th part of duodenum in the hierarchical task analysis tree for pancreticoduodentomy.

D. Removing the Tumor and Pancreaticojejunostomy

At this point in the procedure the pancreas is visible. Before removing the tumor of the pancreas, a small surgical drain is inserted under the neck of the pancreas [10] which is removed 4 days after the (Fig. 1d) [11]. Before, removing the tumor, surgeon needs to perform careful examination to assess the tumor spread. This requires expert camera navigation skills. Removal of the tumor at the pancreas is done by an electrocautery (Fig. 1d) [10]. Next part in the procedure is to create a pancreaticojejunostomy, connecting the left over pancreas to the jejunum (Fig. 1d). This is performed by taking the inner part of the pancreas that is left behind from the electrocautery. Then the inner part of the pancreas is sutured to the inner part of the jejunum (Fig. 1d) [10].

E. Hepaticojejunostomy and Duodenojejunostomy

The outer part of the pancreas and the outer part of jejunum is sutured together. A hepaticojejunostomy is then created by suturing the hepatic duct to the jejunum (Fig. 1e) [10]. The last step in the surgical procedure is to create a duodenojejunostomy. This is performed by taking the outer layer of the jejunum and suturing it to the first portion of the duodenum (Fig. 1e) [10]. Next, an incision into the intestine and the jejunum is performed. After duodenojejunostomy, gallbladder needs to be removed due to the fact that bile duct is attached to the jejunum. The bile flows from bile duct to jejunum and gall bladder no longer stores bile so the function to store the bile's is lost (Fig. 1e).

Next section introduces implementation of surgical simulation development components of VPanSS.

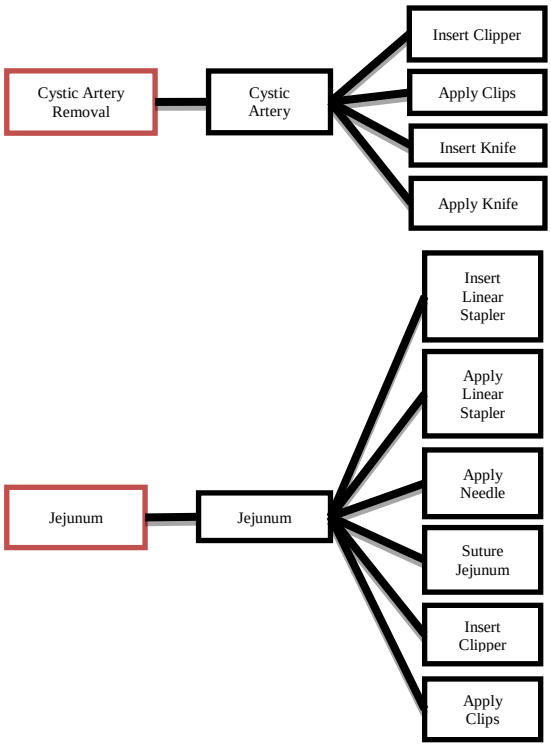

Fig. 1c Subtasks cystic artery removal and jejunum in the hierarchical task analysis tree for pancreticoduodentomy.

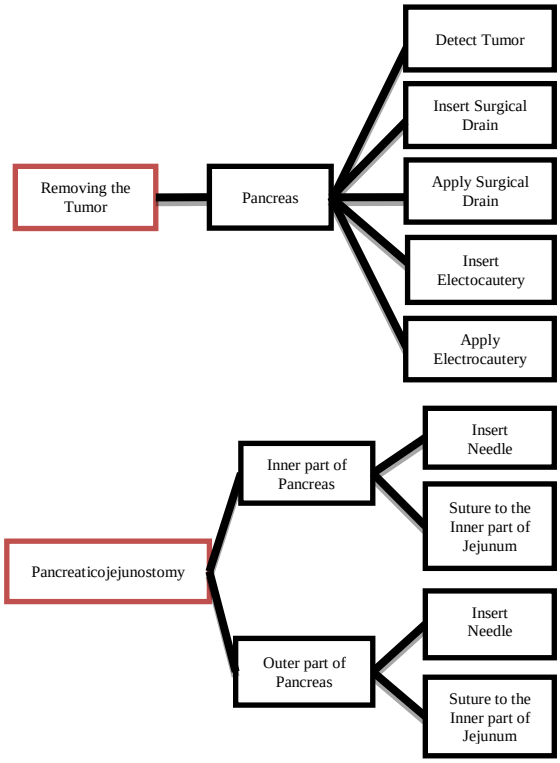

Fig. 1d Subtasks removing the tumor and pancreaticojejunostomy in the hierarchical task analysis tree for pancreticoduodentomy.

III. VPanSS Design

Navigation of the laparoscopic camera needs to be performed manually and requires high cognitive function and visuospatial skills [13]. These skills are essential for navigating the laparoscopic camera in a complex environment like human body. After locating the pancreas, the surgeon needs to re-examine the pancreas and the adjacent tissues to ensure that the tumor has not spread. For this, surgeons need to navigate around and explore the pancreas for any spread. This task requires advanced skills for camera manipulation to find correct position and orientation as indicated in section E.

In order for surgeons to practice navigation of the camera, we created a virtual abdominal environment for a preliminary task of reaching the pancreas. A novel collision detection algorithm called Balls Hierarchy by Kockara et al. [14] has been implemented and integrated in to the Π-SoFMIS. Π-SoFMIS, Halic et al. [15], is a framework developed for creating 3D surgical simulations in a browser environment. Next sections will introduce the architecture and components of VPanSS.

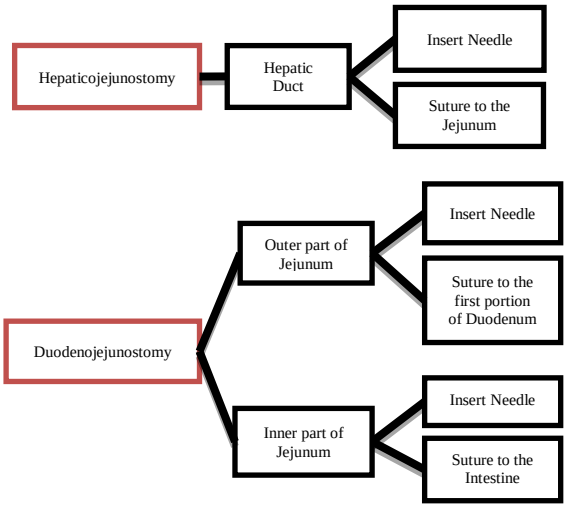

Fig. 1e Subtasks hepaticojejunostomy and duodenojejustomy in the hierarchical task analysis tree for pancreticoduodentomy

A. Simulation Components in VPanSS

Following figure (Fig. 2) shows the process flow and the components of the VPanSS; 1.Razer Hydra (motion controller), 2.Collision Detection Module, 3.WebGL Rendering, 4.3D Import module for JSON files. Razer Hydra was integrated to capture six degrees motion for the laparoscopic camera movement. The movement of the tool is passed into collision detection module. In collision detection module, the motion of the camera is checked against the virtual organs before the scene is drawn. WebGL component is essential to rendering 3D models in web browsers. We load the geometry of the organs by using JSON files. The import and export module of Π-SoFMIS is used for loading JSON file for each virtual organ along with their texture data. The geometry is both used by rendering and collision detection module in the VPanSS.

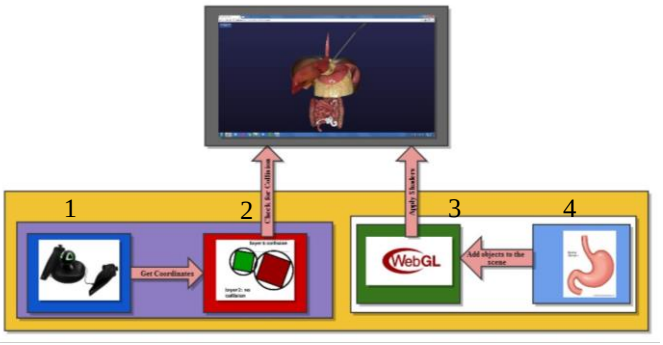

Fig. 2 VPanSS Components

The figure (Fig. 3) illustrates the browser plug-in developed for integrating Razer Hydra motion controller in to VPanSS. The Razer Hydra provides translational and orientation (such as pitch, roll and yaw) motion with magnetic tracking. For interfacing the Razer Hydra with our simulator, a plug–in is necessary for native access to the hardware in each and every simulation execution frame as web browsers and JavaScript do not have access to direct hardware control. The figure (Fig. 3) represents the plug-in architecture for any device including force feedback devices.

The Plug-in is initialized only once by web browser during the start-up of the simulation. In this phase, Web-browser directly calls device specific initialization routines. Once the plug-in is initialized, all functions and plug-in properties can then be used through a JavaScript plug-in object. During VPanSS, the hardware information can be directly accessed and device runs separately at a different thread. This allows achieving high update rates for the plug-in.

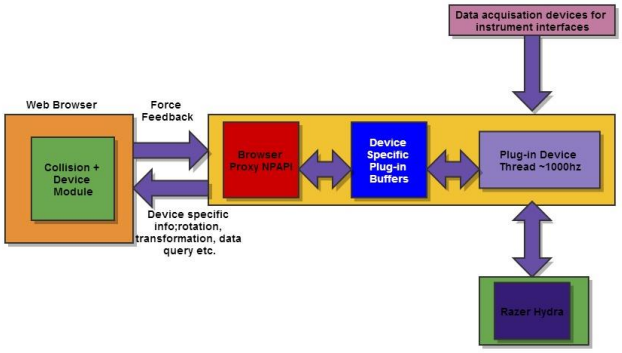

Fig. 3 Plug-in architecture.

B. Π-SoFMIS

The framework Π-SoFMIS [15] has been used for the development of VPanSS. This framework is entirely built with JavaScript and WebGL allowing real-time, hardware independent, portable and accessible 3D visualization platform without any installation. The framework requires only web browsers. The rendering routine utilizes WebGL for realistic rendering of anatomical structures with shaders using multiple maps such as; bump, specular, displacement and alpha. WebGL is based on OpenGL Embedded Systems 2.0 and allows direct access to the graphical processing unit (GPU) within a web browser. This allows efficient generation of 3D interactive applications on any platform capable of running a

web browser. The framework is built with JavaScript, and as a result objects are not tied to any browser specific implementation. This provides true platform independence. Implementation of the framework Π-SoFMIS is completely based on prototyping. With prototyping, objects or functions can be augmented to create new objects. This allows design and implementation of object oriented hierarchy that is used throughout the framework (Fig. 4) [15].

Π- SoFMIS uses JSON (JavaScript Object Notation) format to import 3D geometry. This standardized format makes importing and parsing files straightforward. In addition, JSON is human readable format and allows extensions. Any .OBJ or .3DS 3D object files can easily be converted to JSON format. We implemented our own module for this conversion from .OBJ to JSON format (Fig. 4).

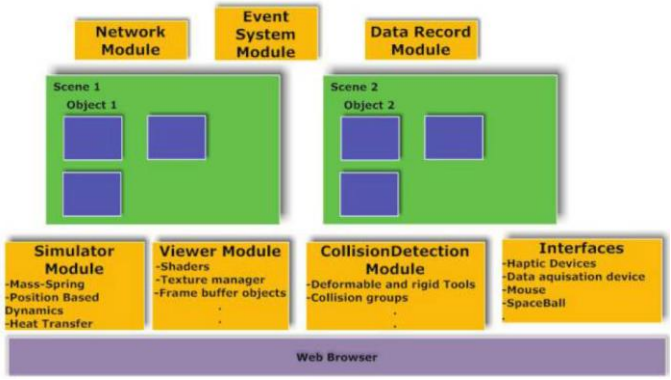

Fig. 4 Π-SoFMIS modules [15]

C. Collision Detection

Contact/Collision detection is an essential part of VPanSS during the camera navigation task. Virtual camera is manipulated with surgeon's hand motions that are tracked with Razer Hyrda game controller. During the navigation task, camera penetration to virtual organs or tissues must be avoided in real-time at all times. For this real-time collision detection purpose, a soft kinetic data structure called Balls Hierarchy was utilized to detect any contact of virtual camera (Fig. 6). The employed soft kinetic data structure, by Kockara et. al[14], dynamically tracks proximities of moving objects and those objects' deformations. The technique handles both broad and narrow phase collision detections, where a spanner tree of hierarchy is being created for each vertex of the 3D organs' meshes and dynamically updated in the case of deformations. Having dynamic trees minimizes recalculation time for deformable bodies.

Construction of the Balls Hierarchy (BH) starts with a random node. It checks if there are any other nodes within the minimum distance, *minDistance* = ($r^* \xi^{level}$). In the formula, *r* represents the preset radius, ξ represents the expansion ratio, which must be greater than 1. *level* refers to the level in the hierarchy. Notice that level expands exponentially from lower levels to higher levels. This provides multi resolution behavior of proximities. Thus, the hierarchy provides early rejection mechanism for far away objects. However, once objects come closer, finer representation of the objects are hold in lower levels. Bounding volume hierarchies are one of the most effective methods for collision detection in virtual scenes. However, they are not suitable for deformable objects' collision since they require excessive hierarchy update operations. This makes them unsuitable for deformable bodies' collisions detections. BH overcomes this problem.

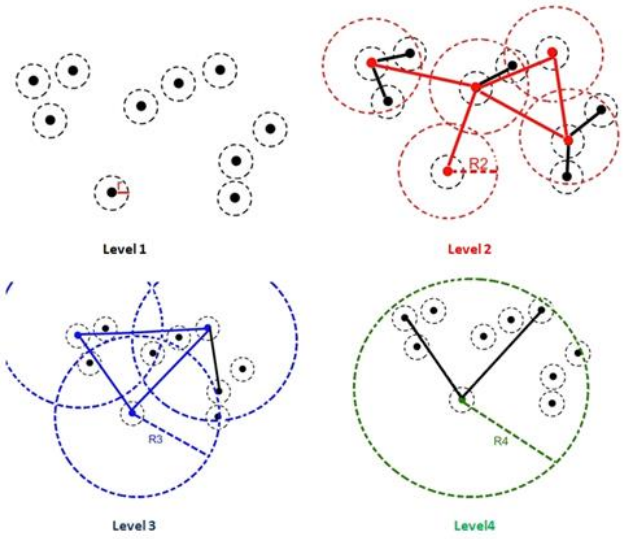

Fig. 5 Hierarchy construction steps

Initially, all the nodes are placed under the level 0 that is specified as the root; no other nodes should have nodes within the minimum distance. By using the minimum distance formula, *minDistance*, all nodes are included at the level 1 either as a parent or as a child node. The same step is repeated for all root nodes for the next levels. Parent nodes are grouped into larger hierarchies until one root node is left. Any change in the positions of the nodes requires an update (e.g. promotion or demotion) in the hierarchy tree. Promotion is leveling up a point whereas demotion is leveling down a point. For deformable structures, only the corresponding section of the BH for moving nodes are recomputed (promoted or demoted) as opposed to all points in the hierarchy. This reduces the computation complexity from *n* to *log (k)* where *k* is the number of vertices affected by motion (e.g. deformation). This behavior makes the BH highly dynamic and adaptive to deformations.

The figure (Fig. 5) illustrates BH construction steps for 10 points. In this illustration, hierarchy is consisting of 4 levels. Each level of the hierarchy is represented in different colors; black, red, blue, and green colors respectively. Dashed circles represent radius-covering at the level. In the first level, original data points exist with minimum radius *r*. Since minimum distance between closest color pairs is 1, *r* is assigned as 1. As seen from the first level, since no other ball center is covered by any other ball, all survive (exist in the level). In the second level, only 5 points survive, since other points are covered by survivors with radius in the second level, R2. Non-survivor points become children of survivors. This relation is represented by child edges which are illustrated as steady black lines. R2 is expanded from minimum radius by expansion ratio

(ξ); thus, $R2 = r\,\xi^{1}$. Superscript *1* represents level difference between level 2 and level 1. Steady red lines in the second level represent neighbor edges which represent neighbor relations between any two survivor points ($N(v^i) = \{u^i \epsilon B_i, |u^i v^i| \leqslant \eta r \xi^{i-1}\}$). Two survivors are neighbors if and only if distance between them is smaller than or equal to the neighbor coefficient (η) times radius at the level (R2). In the third level, there exist only three balls (blue) with radius $R3 = \xi^2 r$, and a single non-survivor which becomes a child. Now, there are 3 neighbors (blue lines) in the third level. In level 4, there is only a single survivor (green) with two children. As indicated, this survivor is called root and covers all existing points. Notice that in order a non-survivor point to become a child of a survivor point, it needs to be a neighbor of the survivor point in the previous level. Once hierarchy tree is constructed, balls hierarchy keeps hierarchical representation of approximate shortest paths among all the existing points. Hierarchy construction time is *O(nlogn)* where n in number of vertices in 3D models. Hierarchy update for deformations takes *O(log n)* if all the vertices are moving. However, for local deformations computation takes a constant time *O(log k)* where *k* is the number of nodes moved during deformation. Algorithm is summarized with pseudo codes in Table-2, Table-3 and Tablet-4 respectively.

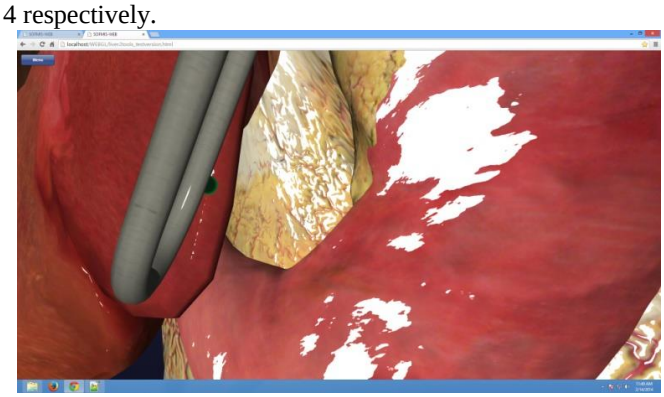

Fig. 6 Scene tool-tissue interaction with colliding point visualized

TABLE 2 COLLISION DETECTION PSEUDO CODE

```
Building the hierarchy
BUILD H
Outputs:
      data, points
      R, radius at this level
      BH, Balls Heirachy
      Node, has one root and any amount of child points
Methods:
      BuildNextLevel(),builds next level of hierarchy
begin
      for every point in data
      mark point as a node root and place in BH level 0
      BuildNextLevel()
end

Building the Next level
Outputs:
      Level-1, structure of points in previous level
      Node, has one root and any amount of child points

Methods:
      BuildNextLevel(), builds next level of hierarchy
      CombineNodes(x,y)
begin
mark all nodes in Level-1 unvisited
      for every unvisited Node x in Level-1
      mark x as visited
            for every unvisited Node y in Level-1
                  if distance(x,y) < radius
                        CombineNodes(x ,y), place in Level
if there are more than one node in this level
      BuildNextLevel()
end

Check Collision
Outputs:
      BH1
      BH2

begin
start at highest level in bh1 and bh2
      while Nodes collide
            reduce level of hierarchy with more levels
            if level == 0
                  return COLLISION, Π-SoFMIS calculates
                  the physics based response
return NOCOLLISION
end
```

TABLE 3 UPDATE PROCEDURE PSEUDO CODE

```
Update Procedure
Outputs:
      Point x, point to be changed
      BH, Balls hierarchy
Methods:
      Insert(point, BH), inserts a point into the BH
      UpdateAllChildren(root, BH), performs the update
procedure on all children of a root node
begin
Remove Point x from BH
      if Point x is root
            UpdateAllChildren(x,BH)
            Insert(Point x, BH)
      else
            Insert(Point x)
 end
```

TABLE 4 INSERT PROCEDURE PSEUDO CODE

```
Insert Procedure
Outputs:
      Point x, point to be changed
      BH, Balls hierarchy
Methods:
      Distance(point,point), gives the distance between
      2 points
      Add(root, point), adds x as a child node to root,
      also if any children of root are root, nodes will
      attempt to add x to them

Variables:
      level, every levels has root nodes and radius
begin
level = Highest level in BH
      for root nodes in level
            if distance(root, x) < level.radius
                  add(root,x)
            else
                  make x root node and add to BH
 end
```

In this implementation, the hierarchical structure is used. Instead of checking every point, BH checks for just the parent of the level. When a collision is not detected on the top level, BH early rejects collisions so that broad phase collision detection is handled. If there is a collision at the bottom level, Π-SoFMIS generates a proper physics based response for deformable bodies. This corresponds to narrow phase collision detection.

D. VPanSS Benchmark

Three different computers and three different web-browsers were used to collect the Frame per Seconds (FPS) for rendering and collision to understand performance of VPanSS. The computers had the following specifications; the first desktop computer, referred as COMP1, had an Intel i7-3820 CPU with 3.60 GHz, an 8 GB memory and GeForce GTX 550 Ti version 311.06 graphics card with Windows 8. The second desktop computer, referred to COMP2 had Intel i7-3770 CPU with 3.40 GHz, a 16 GB memory and GeForce GT640 version 327.23 graphics card with 64-bit, Windows 7. The third laptop computer, referred to COMP3, had Intel Core i7-3630QM CPU with 2.40 GHz, a 8GB memory and GeForce GTX660M version 334.89 graphics card with 64-bit, Windows 7 Operating System. Three major web browsers; Mozilla Firefox 26.0, Google Chrome 31.0.1650.63m and Chromium 35.0.1865.0(254057) used in benchmark data. The critical components in navigation tasks are contact detection module and visual rendering. The performance data was collected over a 60 seconds time period, and then the average FPS was computed. Three different tests were performed. The first test was performing collision detection without rendering any objects. This was performed to identify collision detection performance. The second test was to measure the rendering performance for the visual scene. The third test was collision detection and rendering of the scene at the same time to determine the simulation overall performance. In the scene, there were 13 objects that consist of 68,883 vertices and 22,961 triangles. In Π-SoFMIS, in order to get accurate frame rates, the timer interval was set to 1 millisecond to get frame rates at ~1000Hz, instead of 16ms(60hz).

The figure (Fig. 7a) shows the FPS calculated when only the collision was being detected. Although all FPS are nearly comparable, in all cases Chromium was superior and Firefox was lower. COM2 had better benchmarks in collision detection without rendering due to the fact that the CPU was superior to COM1 and COM3.

The figure (Fig. 7b) shows the FPS calculated when only the scene is being rendered without collision detection module enabled. The data was collected when the scene was rotating among the Y-axis. In all the cases, Google Chrome was superior, Firefox was lower. Unlike the previous case, Chromium's rendering performance exhibits performance drop around 50-100 FPS with respect to Chrome browser. With the better graphics card, COM1 was faster amongst all browsers in rendering test.

The figure (Fig. 7c) shows the FPS calculated when VPanSS was working with both rendering and collision detection at the same time. In all cases, Firefox had lowest and Chromium had highest rates. COM1 and COM2 had better benchmarks compared to COM3, which was as expected due to its lower hardware specification in regards to the others.

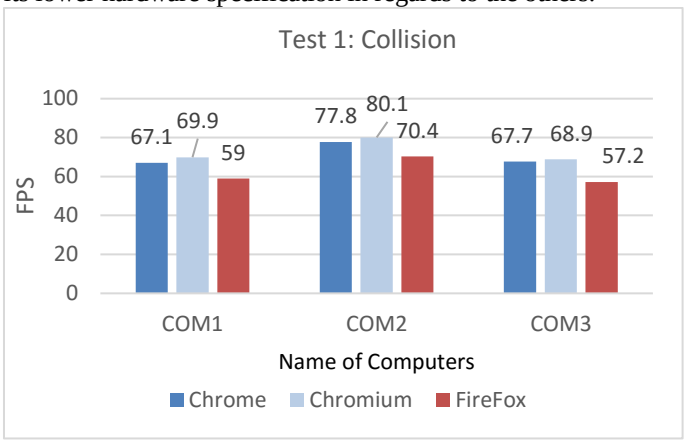

Fig.7a Frame per second when only the collision is being detected

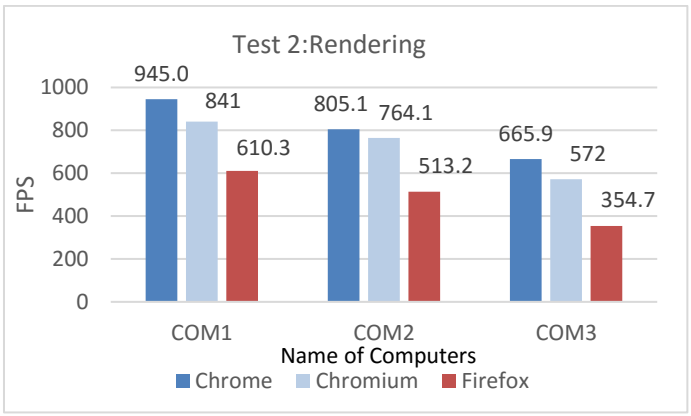

Fig. 7b Frame per second when only the scene is being rendered and movement without collision is being simulated

In all test cases, Firefox attained the lowest performance on three different computers. As observed from the data, performance can noticeably vary with different browser engines. Therefore, we are working on complete multithreaded and GPU based versions for collision detection to take load off of the main thread of the web application [15].

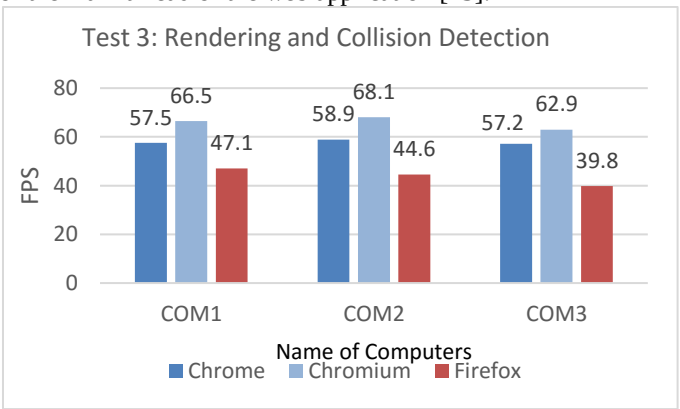

Fig. 7c Frame per second when the simulator was working with rendering and collision detection

E. Camera Navigation Task

As seen in figure (Fig. 8), a navigation task was developed aiming at increasing the visuospatial and handling skills of surgeons. In this task, surgeons are asked to navigate to the pancreas via manipulating a laparoscopic virtual camera. First goal of the tasks is to locate and find the virtual flashing sphere randomly positioned at pancreas. In the second task, the surgeons are asked to navigate camera and get close to the sphere as much as possible while avoiding any collisions to the organs. The amount of contacts between virtual camera and organs is recorded until that task is completed. The task score is computed based on the number of collisions that would reflect the proficiency level of the surgeon.

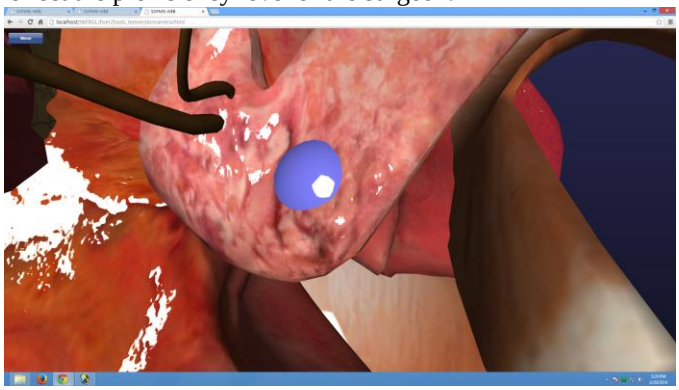

Fig. 8 Flashing target in the simulator

IV. Conclusion

"Whipple" procedure is a complicated surgery with most subtasks that affiliate with adjacent organs that could take up to 11 hours [8]. In such a complex surgery, complications could easily develop if the surgeon lacks enough skills and experience. It is revealed that surgeons and hospitals with less experience have almost four times more operative mortality rate compared to expert surgeons. Although gaining experience is very critical, there is no available training platform for pancreas surgery. Therefore, we developed VPanSS; a preliminary pancreas virtual simulation. VPanSS has the camera navigation task where the surgeons could practice and get their training without any risk. VPanSS is a platform independent, portable, accessible real-time web based simulation. We tested our simulation components on three different computers using Chrome, Chromium, and Firefox browsers to verify its portability and accessibility. Test results indicate that simulation successfully achieves real-time rates with standard computers. At present VPanSS is a training platform but it could be also used to assess the surgeons' skill levels for the camera navigation in "Whipple" surgery. Ongoing development is focused on validation of the simulator with different surgeons and to get their objective and subjective feedback.

References

[1] J. F. Tseng, P. W. Pisters, J. E. Lee, H. Wang, H. F. Gomez, C. C. Sun, and D. B. Evans, "The learning curve in pancreatic surgery," *Surgery*, vol. 141, no. 5, pp. 694–701, 2007.

[2] J. D. Birkmeyer, A. E. Siewers, E. V. Finlayson, T. A. Stukel, F. L. Lucas, I. Batista, H. G. Welch, and D. E. Wennberg, "Hospital volume and surgical mortality in the United States," *N. Engl. J. Med.*, vol. 346, no. 15, pp. 1128–1137, 2002.

[3] P. Watanapa and R. C. N. Williamson, "Surgical palliation for pancreatic cancer: developments during the past two decades," *Br. J. Surg.*, vol. 79, no. 1, pp. 8–20, 1992.

[4] B. W. Kuvshinoff and M. P. Bryer, "Treatment of resectable and locally advanced pancreatic cancer," *Cancer Control*, vol. 7, no. 5, pp. 428–436, 2000.

[5] J. A. Sosa, H. M. Bowman, T. A. Gordon, E. B. Bass, C. J. Yeo, K. D. Lillemoe, H. A. Pitt, J. M. Tielsch, and J. L. Cameron, "Importance of hospital volume in the overall management of pancreatic cancer.," *Ann. Surg.*, vol. 228, no. 3, p. 429, 1998.

[6] U. Giger, I. Frésard, A. Häfliger, M. Bergmann, and L. Krähenbühl, "Laparoscopic training on Thiel human cadavers: a model to teach advanced laparoscopic procedures," *Surgical endoscopy*, vol. 22, no. 4, pp. 901–906, 2008.

[7] A. A. Gumbs, B. Gayet, and J. P. Hoffman, "Laparoscopic Whipple procedure with a two-layered pancreatojejunostomy," *Surg. Endosc.*, vol. 25, no. 10, pp. 3446–3447, 2011.

[8] M. Gagner and M. Palermo, "Laparoscopic Whipple procedure: review of the literature," *J. Hepatobiliary. Pancreat. Surg.*, vol. 16, no. 6, pp. 726–730, 2009.

[9] C. Palanivelu, R. Shetty, K. Jani, K. Sendhilkumar, P. S. Rajan, and G. S. Maheshkumar, "Laparoscopic distal pancreatectomy," *Surg. Endosc.*, vol. 21, no. 3, pp. 373–377, 2007.

[10] Cameron, J. L., & Sandone, C. "*Atlas of gastrointestinal surgery: Pancreas"*. 2 ed., Vol. 1, pp. 284-305. BC Decker.

[11] G. Srikanth, N. Shetty, and D. Dubey, "Single incision laparoscopic distal pancreatectomy with splenectomy for neuroendocrine tumor of the tail of pancreas," *Journal of minimal access surgery*, vol. 9, no. 3, p. 132, 2013.

[12] T. Mori, N. Abe, M. Sugiyama, and Y. Atomi, "Laparoscopic hepatobiliary and pancreatic surgery: an overview," *Journal of hepato-biliary-pancreatic surgery*, vol. 9, no. 6, pp. 710–722, 2002.

[13] M. G. M. MDl, B. J. Mohler, M. L. Disc, and A. Lefever, "A virtual reality surgical trainer for navigation in laparoscopic surgery," *Medicine Meets Virtual Reality 2001: Outer Space, Inner Space, Virtual Space*, vol. 81, p. 171, 2001.

[14] S. Kockara, M. Mete, V. Yip, B. Lee, and K. Aydin, "A soft kinetic data structure for lesion border detection," *Bioinformatics*, vol. 26, no. 12, pp. i21–i28, 2010.

[15] T. Halic, S. A. Venkata, G. Sankaranarayanan, Z. Lu, W. Ahn, and S. De, "A software framework for multimodal interactive simulations (SoFMIS).," in *MMVR*, 2011, pp. 213–217.