University of Western Ontario

Software Specifications: Camera Simulation Environment

Version 0.02

Author:
Kartik Thakore
kthakore@uwo.ca

Supervisors:
Dr. Hanif Ladak
Dr. Terry Peters

February 1, 2012

Contents

List of Figures

Glossary

Application Server A web service provider that responds to requests from the client. 8, 9, 16

extrinsic Parameters of a camera that define the coordinte system transformations from 3D world coordinates to 3D camera coordinates. Two matrices are provided R and T. 6, 7, 12, 16, 17

Frustum Viewing volume of a camera. 17

HTML5 Is the fifth version of the Hypertext Markup Language that provides many new features such as canvas and web sockets. 8, 15

HTTP HyperText Transport Protocol defines the protocol of transporting documents to a web client. 8

intrinsic A single matrix that contains 5 parameters. The parameters incorporate focal length, image format and principal point.. 6, 7, 12, 16, 17

Javascript A cross platform scripting language that runs on the client side. 8, 16

OpenCV A software library that has a camera calibration component. 8, 9, 12, 17

Pinhole Camera Model The pinhole camera model defines a camera as a mathematical relation between the coordinates of a 3D point, projected onto an image plane. This is also called the Perspective Model. It is a common camera model used in computer vision. 6

SIT SIT: System Integration Testing, tests the overall system via, given inputs and expected outputs. 7

SRS Software/System Requirement Specifications. 15

UAT UAT: User Acceptance Testing, tests the system to ensure the user experience and expectations are met. 7, 10, 18

VR Virtual Reality: is a simulated environment that is implemented to provide a simulated experience of a real world process or activity. 6–8, 12

Web Browser A cross platform software package to view HTTP provided content from service providers. 8

WebGL A web technology being developed by the Khronos Group to bring accelerated graphics to the web browser. 8, 16

Chapter 1

Revisions

Date	Version	Author	Description
January 22 2012	0.01	Kartik Thakore	Initial Specifications Document
Feburary 1 2012	0.02	Karitk Thakore	Incorporating Dr. Ladak's Changes

Chapter 2

Introduction

A core component of Medical Imaging application is the use of computer vision. Camera calibration is a common task required for computer vision. Camera calibration can be a difficult task for researchers without prior graphics background. Cameras have been modeled extensively in literature, and thus can be simulated. A virtual environment provides an activity based learning approach for camera calibration.

2.1 Overview

1. A virtual reality (VR) environment that simulates extrinsic and intrinsic parameters of a camera will provide a sufficient model to allow training on transformation matrices involved with camera calibration.

2. Camera calibration of a simple camera system, performed with prior training in a virtual reality environment, will yield insignificantly different capture camera parameters with real world camera system.

2.2 Background and Significance

Medical Imaging tasks such as image fusion, registration, feature tracking, etc. are built on the understanding of the mathematical concepts of vision systems. Computer vision is an increasing component of image-guided interventions. Moreover it is often difficult for researchers not from a computer vision background to learn quickly. Virtual environments that simulate computer vision systems will both hasten the learning curve and also act as a prototyping environment for new applications of video-based technology in image-guided interventions.

Mathematically, vision systems can be represented as matrices that perform operations on a spatial field. Cameras in medical imaging essentially transform three-dimensional content to two-dimensional planes (Pinhole Camera Model [2]). Thus, locations in 2D images correspond to the position and orientation of objects in the real world.

To be able to capture the position and orientation of objects, a transformation matrix is required from the 2D images. This transformation matrix is acquired by performing camera calibration. Calibration involves acquiring images of known objects with known positions and orientations in the real space. Next, feature points are selected on the 2D image. The selection of feature points are done either manually or using libraries such as OpenCV. The correspondence sets up the transformation between the real world coordinate system and the image coordinates.

Simulation of this process can begin by modeling the camera using the perspective projection model. The model can be represented in matrix notation as:

$$s * \mathbf{p} = \mathbf{A} * [\mathbf{R}|\mathbf{t}] * \mathbf{P} \tag{2.1}$$

where A is defined as the intrinsic matrix, s represents the arbitrary scale factor, p are the 2D image coordinates, P are the corresponding 3D world coordinates, and finally $(R|t)$ are the extrinsic parameters.

The intrinsic matrix can be calculated from camera dependent parameters such as focal length and CCD pixel. Additionally, the extrinsic parameters are user defined such that rotation and translation are selected by the user. Finally, the P world coordinates are simply defined by the user and the scene that will rendered in the virtual reality environment.

A key challenge faced by graduate students in areas such as image-assisted surgery is in learning the calibration process and understanding the sensitivity of the calibration results on estimated parameters. Indeed, experience at the Robart's Research Institute indicates that students can take several months to learn about camera calibration. Simulation will allow for applications such as training and prototyping.

Once the prototype software system is developed, System Integration Testing (SIT) will be performed. Validation will be performed with User Acceptance Testing (UAT) of a selected group of volunteers. This will aid in narrowing down the causes for difficulty with understanding camera calibration.

2.3 Relevance

Camera calibration is a difficult task and a widely recurrent task at Robart's Research Institute in Medical Imaging. Additional camera calibration is a means to an end in most projects and not the focus. The essential tasks involved for performing a camera calibration are similar across projects with potential for standardization and collaboration. In Dr. Peters' laboratory, a considerable amount of effort is placed in camera calibration in several research projects. Each project involves isolated or loosely collaborated work on camera calibration. No software currently exists to help with training of camera collaboration.

Selecting a camera is crucial to a medical imaging application. A virtual environment may help to explore and compare various camera systems based on parameters provided by manufacturers. Allowing researchers to objectively try systems rather then purchasing each system and performing manual calibration. This will facilitate prototyping and will be a boon for researchers in gaining an intuitive understanding into the critical factors of camera calibration and the impact on their application.

2.4 Scope

The scope of this project is primarily to create a virtual environment that can reasonably simulate common cameras used in medical applications. The system will be able to create cameras given parameters that can be used to view a scene and run camera calibration tests on. Once calibrated, the system will allow the user to capture virtual images.

1. Develop a VR environment that can calculate transformation matrices based on given locations, poses, and camera parameters.

2. Implement a VR environment that simulates camera calibration within user specified scenes. Develop an online tool that can be accessed by a wide audience in the medical imaging field.

2.4.1 Pilot Scope

The scope of this pilot is to explore and define (from the prespective of a new researcher) more deeply the problems with camera calibration, and a possible solution.

2.5 Technical Overview

2.5.1 User Interface

One of the initial aims of this project was to provide a easily distributed user interface. A web interface is a proven technology in providing distributed interfaces. The user interface provided by this system will be designed on a HTML5/WebGL project that will work over pre-specified Web Browser. WebGL sits on top of HTML5's canvas feature and will provide 3D rendering capabilities on enabled Web Browser. Most user demographics are familiar with using a Web Browser, which makes this an ideal option.

WebGL will be written in the Javascript language that is a platform provided by most modern Web Browser.

The Web Browser is also very portable compared to other solutions, as most modern computer systems come with one installed. The need to have a WebGL enabled Web Browser may hinder the proliferation of the system. However most new versions of Web Browser are coming with WebGL enabled and it is a emerging trend.

The Web Browser finally also comes at no additional cost to the user.

An alternative option involves creating custom distribution and front end software (similar to a Web Browser). This is not feasible with the time frame for a Masters project.

2.5.2 Application Server

An Application Server will be built to provide the Javascript and HTML5 documents to the Web Browser. Additionally, the server will utilize middleware platforms that help to scale (ensures the Application Server responds to requests from multiple Web Browsers).

2.5.2.1 HTTP Server

The HTTP Server will be responsible for responding to requests sent by Web Browser clients. The key requirement required of the HTTP Server is to scale reasonably with an increase in demands. Most of the computational expense will be done on the Web Browser client, due to the way WebGL works. A scalable HTTP Server will ensure that the response for the WebGL and HTML5 documents are provided in a timely manner. For this reason the Nginx HTTP Server and a Starman/PSGI middleware server will be used.

A middleware server allows multiple Application Server to be initiated and used in parallel for times of load. This facilitates the reliability and scalability of the system.

2.5.2.2 Dispatch/Controller

The user will be using the system to do camera calibration that will require a interface between the frontend and OpenCV features. A controller will be used to define to protocol by which the frontend will communicate with the OpenCV Server.

2.5.2.3 OpenCV Server

This server will provide the Application Server with OpenCV features such as camera calibration and re-projection of calibration grids.

Chapter 3

User Specifications

This chapter defines the user of this system and their specifications. Defining the user ensure that the system can be designed with a UAT goal in mind.

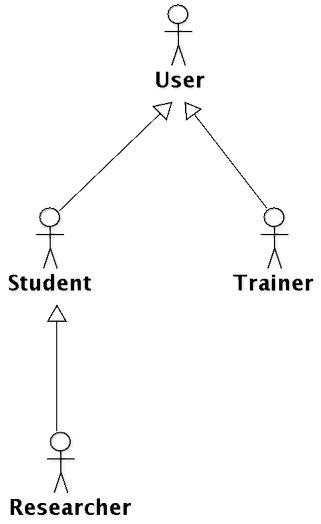

Figure 3.1: Users

3.1 Users

The system user is a generalization of several other users as depicted in the use case diagram 3.1.

3.1.1 Student

The student user is the primary user of the system. This system is essentially made to facilitate the understanding of the student user. Student users are defined as any user that needs to learn and work with camera transformation.

3.1.1.1 Researcher

The student user can also be extended as a researcher who needs a deeper understanding of camera calibration for a research project. These project involve camera calibration as a critical component in the implementation.

3.1.2 Trainer

Another user that may use the system is the trainer. Trainer users use the system as a guide for students and researchers. The direct requirements of the trainer user needs to be captured at a later date.

3.2 Use Cases

Use Cases define the process the system will take to meet the needs of the users. Each use case is written in a list format showing what the user will see and react with during the specific portion of the system.

3.2.0.1 Camera Calibration

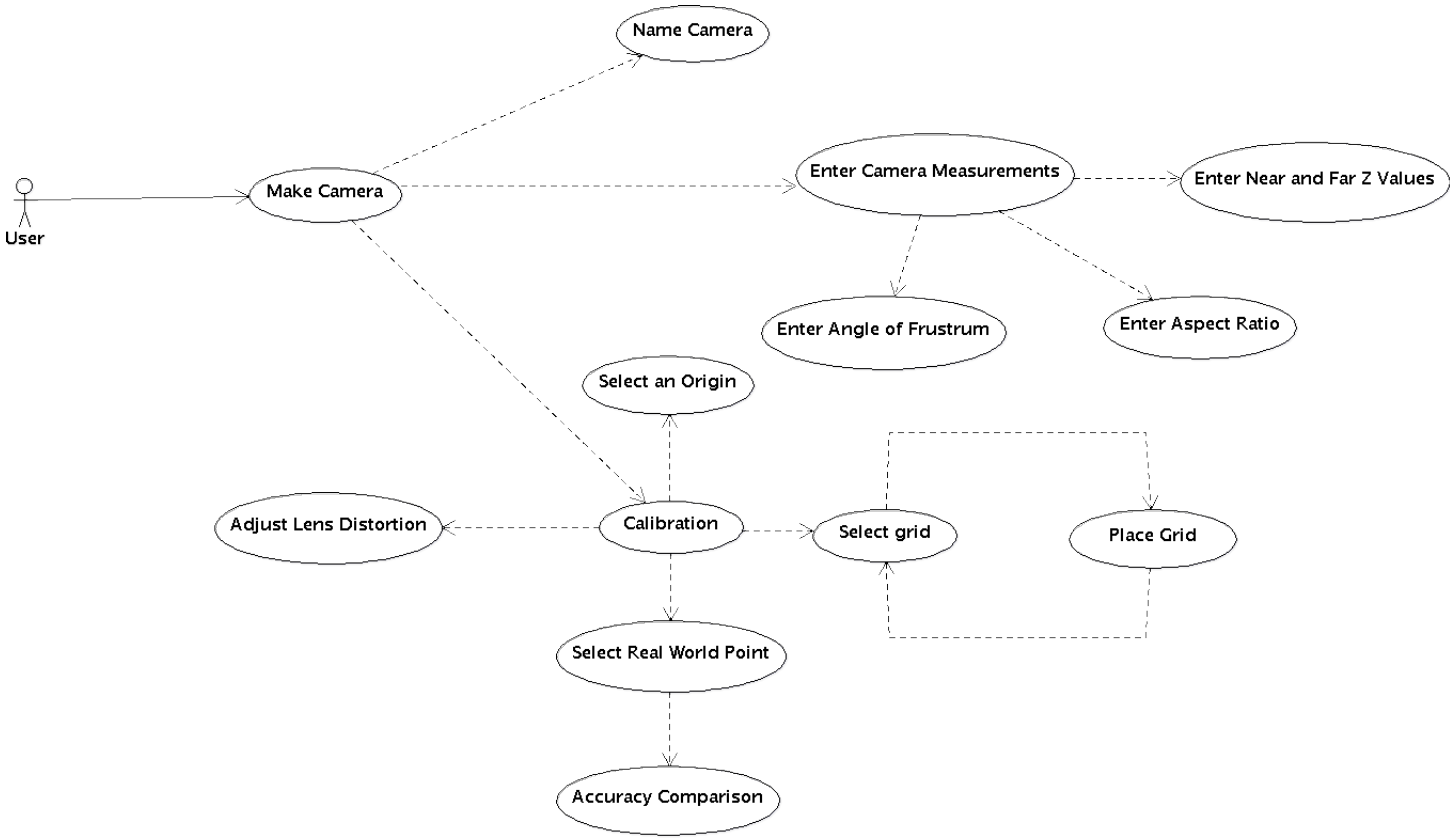

Figure 3.2: Generic User Cases

As described in 3.2, users can perform the camera calibration procedure as defined in Chapter 2 of [1]. All users of the system can perform a camera calibration procedure.

The first step involves creating a camera by with specified measurements. Users in future versions may select existing cameras that have been validated and stored on the system.

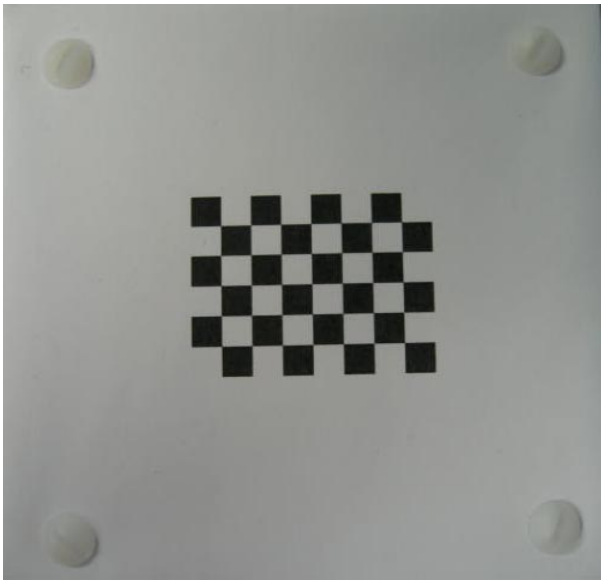

Figure 3.3: Calibration Grid –[1]

Using the camera the User can then select an origin point and place calibration grids from 3.3 within the VR. The user then capture the placement of the calibration grids as images. The images are analyzed for lens distortion (provided by OpenCV).

3.2.0.2 Constructing Transforms

The user can apply the camera calibration by mapping the calibrated camera's origin point to a real world coordinate system. Further analysis of the use case is required to decide between a haptic arm or mouse solution.

This transforms between the calibrated transforms and the real world application (depicted by $T_{R->S}$ in 3.4). Finally by adjusting the lens distortion parameters the user can compare the accuracy at real world points between sets of distortion parameters.

3.2.0.3 Comparing Camera Systems

The Researcher user can perform calibration on two cameras and compare their accuracy models, to decided if they are suitable for their application. This use case is depicted in figure 3.5.

3.2.0.4 Training

Training involves two users. The Trainer can create a training session with a specified calibration procedure (camera, distortions, grid ... etc). Then the Trainer can send this session to a Student.

The Student will follow the Training Session and work towards a specified goal by the Trainer. The Student can also leave their feedback on completion that the Trainer can review, as described on 3.6

The Student completes the training session by finishing camera calibration and sending back the camera distortion parameters, extrinsic and intrinsic values to the trainer. Additionally the student

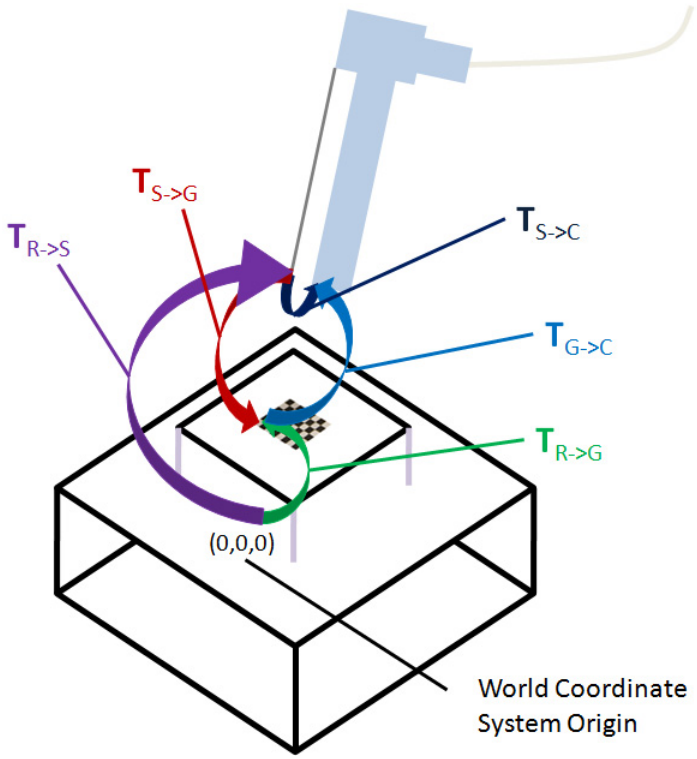

Figure 3.4: Transformations involved during Calibration –[1]

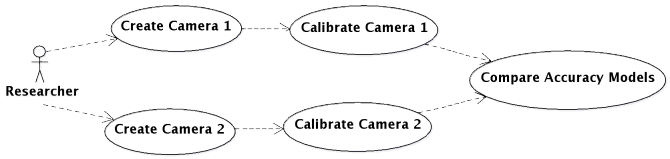

Figure 3.5: Researcher Use Cases

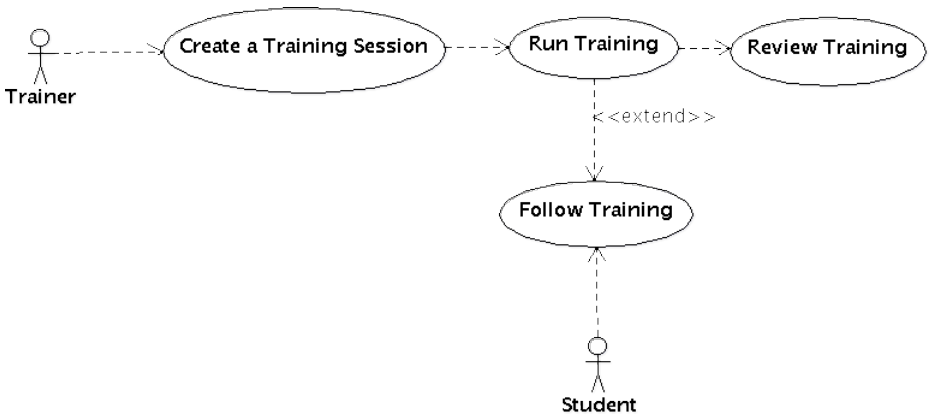

Figure 3.6: Generic Training User Cases

can record the difficulty level of the steps of camera calibration. This feedback can be reviewed by the trainer.

Chapter 4

System Requirement Specifications

The follow System Requirement Specifications (SRS) serve to outline a set of deliverables or goals for the prototyping period of analysis, design, and implementation. The preliminary specifications are defined here to provide a starting point for prototyping. The initial round of testing on the prototype will help to clarify, extend and solidify the next iteration of specifications. This document is meant to modified and iterated.

4.1 User Interface

The user interface will be designed with standard HTML5 provided components for capturing user inputs, and providing feedback. These elements are well known and understood by a wide variety of users, reducing the steepness of the learning curve.

4.1.1 Interface Overview

There are no specific requirements for the user interface as of yet; however a general understanding of the screens has been captured as described in 4.1.

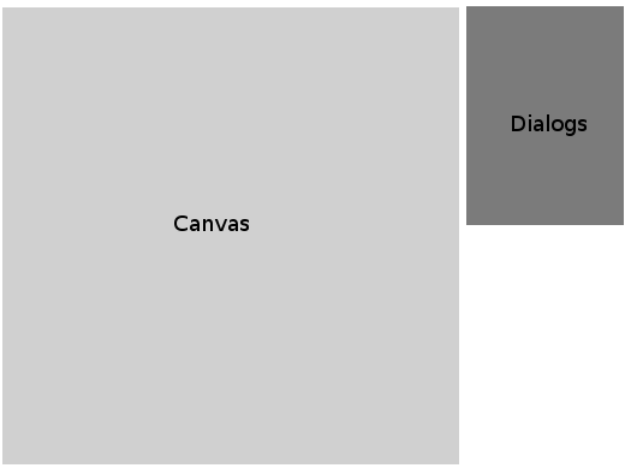

Figure 4.1: Interface Overview

The interface overview can be broken down into two sections. The canvas and the options menu. All 'work' related with camera calibration will be visualized in the canvas section of the interface.

All inputs will be captured in the dialogs section of the interface. Any additional outputs will be show via the dialogs sections also.

4.1.1.1 Canvas

The canvas will be a WebGL environment which which visualize the 3D state of the client.

4.2 Data Models

The data models of the system are software constructs that the client holds, containing groups of related data. The models may or may not be sent to the Application Server for storage.

4.2.1 Storage

Models will be stored as a JSON (Javascript Object Notation) within the Javascript cache of the client. Client side storage will be temporary and non persistent per user session.

4.2.2 Known Models

The current known models are:

4.2.2.1 Camera

Attributes include:

1. name: human readable content to allow comparison.
2. measurements: measured values as described in the use case 3.2
3. calibration session: status of the current calibration procedure. Including the extrinsic, intrinsic and distortion values.

4.2.2.2 Calibration Session

Attributes include:

1. name: human readable content to allow comparison.
2. steps: predetermine step in calibration. This will be a generic template, until various procedures are designed by a Trainer.
3. matrices: extrinsic, intrinsic and distortion values
4. coordinates system: coordinate systems associated with this calibration

4.2.2.3 Training Session

Attributes include:

1. camera: camera to work with (specified by Trainer)
2. student: name of the student currently working on the calibration, including email.
3. calibration feedback: comments of the student and difficulty
4. trainer: email of the trainer

4.2.2.4 Student Feedback

Attributes include:

1. difficulty: rating
2. description: written comments

4.3 Functional Requirements

4.3.1 Create Camera

User will complete the Camera data model described earlier.

4.3.2 Calibration

User will select a calibration grid and pose it 6 times within the 3D canvas. User will use an OpenCV interface to retrieve distortion parameters. User will analyse and derive the extrinsic and intrinsic values of the parameters.

4.3.3 Accuracy Modeling

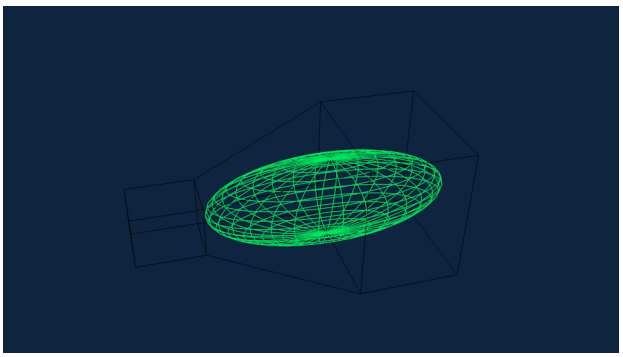

Figure 4.2: Visualising the accuracy model within the frustum of a Camera

User will adjust distortion project and project the selected calibration grids to selected poses. System will calculate difference between points on the actual and projected grid. The accuracy can also be taken with placing points within the viewing range (Frustum) of each camera and calculating difference between ideal (no distortions) and changing distortions.

User can also perform accuracy modeling between two cameras to decide which one to pick. The accuracy will be visualized to show areas of acceptable accuracy that the user picks. A potential visualization concept is depicted in the figure 4.2.

4.3.4 Training

The Trainer will provide a student's email, and decide on steps for training on (changes in data models). The steps are created to watch for changes in the data models(such as values confirmed for extrinsic matrix). These are essentially steps that can be watched and associated with student's tasks within the larger activity of camera calibration.

The trainer can then send a hyper text link via email to the student. The student will go to the link and follow the training steps while providing the feedback. The feedback will collect the student's difficulty and provide a report to the trainer.

The trainer can then review each step and then work with the Student to eliminate learning problems within identified difficult areas.

4.4 Constraints

4.4.1 Reliability

Reliability constraints are the same as expected of other web tools available. Industry standards of scalability are deployed into the system. Further testing will need to be performed to ensure expected reliability. This will be performed in UAT.

4.4.2 Scalability

Scalability constraints are the same as expected of other web tools available. Industry standards of scalability are deployed into the system.

4.4.3 Performance

Performance constraints are lax, as this is a prototype and performance profiles are not available.

4.4.4 Security

Currently there are no security constraints known.

Bibliography

[1] Carling L. Cheung, *Fusion of stereoscopic video and ultrasound for laparoscopic partial nephrectomy*, School of Graduate and Postdoctoral Studies, University of Western Ontario, London, Ont., 2010.

[2] David Forsyth and Jean Ponce, *Computer vision: a modern approach*, Prentice Hall, Upper Saddle River, N.J., 2003.