Multi-Camera Stereoscopic Vision: Design Document

Erin Sullens, John Miller, Sam Schultz

CS461
Fall 2016
Group 54
Sponsor: Kevin McGrath
Group Name: ImMaculaTe Vision

Abstract

This design document describes the software life cycle and the implementation process of building the mobile and desktop applications for our project, Multi-Camera Stereoscopic Vision. The technologies that we are planning to use and how they will be utilized in each processing component of the complete system, are described in detail.

Kevin McGrath Date

John Miller Date

Sam Schultz Date

Erin Sullens Date

Table of Contents

1 Introduction

1.1 Purpose

The purpose of this SDD is to plan out how all project entities work together and how we will develop the project. The intended audience of this document is the sponsor of this project, and the development team.

1.2 Scope

This document describes the software design patterns behind the Multi-Camera Stereoscopic Vision. This project will be used to generate a stereoscopic 3D video from two videos taken from cameras mounted at a known distance apart. This project is intended for use by any user who is interested in generating stereoscopic 3D videos and has the required and compatible hardware to do so.

1.3 Overview

This document covers all software related entities in the project, how they work together, and how the development team will develop the project. The software design document is organized according to the IEEE SDD format. The body of this document consists identifying the stakeholders and their concerns, and the different viewpoints including, Context, Composition, Dependency, Patterns use, Interface, Structure, Interaction, State Dynamics, and Algorithms.

1.4 Authorship

1.4.1 Erin is responsible for sections: 2.3.1, 2.3.2, 2.3.3, 2.4, 2.7.3, 2.7.5, 2.8.1, 2.8.2, 2.8.3, 2.9, 2.10.1, 2.10.2, 2.10.3

1.4.2 John is responsible for sections: 2.2, 2.5, 2.6, 2.7.1, 2.8.1

1.4.3 Sam is responsible for sections: 2.3.4, 2.3.5, 2.4, 2.7.2, 2.7.4, 2.8.4, 2.8.5, 2.9, 2.10.4, 2.10.5

1.5 References

[1] kjkjava, 'kjkjava/garmin-connect-export', 2015, December 22. [Online]. Available: https://github.com/kjkjava/garmin-connect-export/blob/master/gcexport.py. [Accessed: 12- Nov- 2016].

[2] NGHIAH012, 'Simple Video Stabilization Using OpenCV', 2014. [Online]. Availeble: http://nghiaho.com/?p=2093. [Accessed: 2- Nov- 2016].

[3] Rosebrock, Adrian, 'Basic Image Manipulations in Python and OpenCV: Resizing (scaling), Rotating, and Cropping'. [Online]. Available: http://www.pyimagesearch.com/2014/01/20/basic-image-manipulations-in-python-and-opencv-resizing-scaling-rotating-and-cropping/. [Accessed: 12- Nov- 2016].

[4] 'OpenCV: cv::stereo::StereoBinarySGBM Class Reference', Docs.opencv.org, 2016. [Online]. Available: http://docs.opencv.org/trunk/d1/d9f/classcv$_{11}stereo_{11}StereoBinarySGBM.html.[Accessed : 14 - Nov - 2016]$.

[5] 'What is stereoscopic 3D?'. [Online]. Available: https://www.thefoundry.co.uk/products/ocula/about-stereoscopic-3d/. [Accessed: 30- Nov- 2016].

[6] 'Create a UML Diagram'. [Online]. Available: https://www.gliffy.com/uses/uml-software/. [Accessed: 30- Nov- 2016].

[7] 'Moqups · online mockups made simple]. in Moqups. [Online]. Available: https://moqups.com/. [Accessed: Nov. 30, 2016].

1.6 Glossary

1.6.1 Stereoscopic 3D video: Using two video files that are taken from slightly different angles in order to produce a feeling of depth. The left video is viewed from the left eye, and the right video is viewed from the right eye [5].

1.6.2 SDD: Software Design Document.

1.6.3 Application: Mobile application (Apple, or Android) and desktop application (Windows, Mac, or Linux).

1.6.4 MVC: Model View Controller.

2 Body

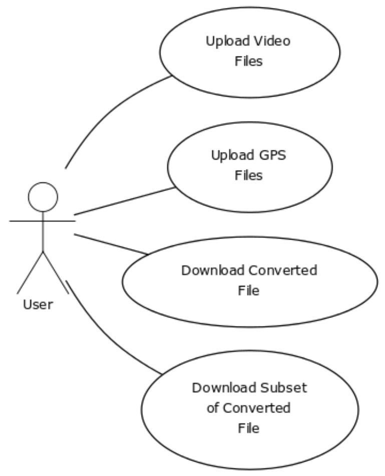
2.1 Stakeholders and their concerns

Kevin McGrath, the sponsor of the project, is the only stakeholder. His only concern is being able to capture the videos in all weather conditions (rain, snow, high winds).

2.2 Context Viewpoint

The system will interact with the user through multiple application user interfaces. The user will upload the video files to be converted to stereoscopic format. The user will have access to previously converted video files available for download. The user will have the ability to select a sub-section of the converted file for download.

Use Case Diagram depicting how the user interacts with the applications:



2.3 Composition Viewpoint

The system will be composed of 5 different processes. These include parsing the BLOB files, stabilizing the videos, cropping the videos, correlating the frames of the two videos, and the conversion of the two video files into one stereoscopic 3D video.

2.3.1 Parsing the BLOB file

The role of this process is to extract GPS timestamp data from the BLOB file provided by the camera, so that we can correlate the frames from the two cameras. We found some parsers online, including a Garmin parser on GitHub called Garmin-connect-export [1]. We are unsure if this parser will work, but if need be, we will use reverse-engineering to parse the BLOB files.

2.3.2 Stabilizing the videos

The role of this process is to take the two videos as input, and output two stabilized videos. For this process, we found functions in the OpenCV library that we will utilize [2].

2.3.3 Cropping the videos

The role of this process is to crop the videos so that both videos have equal field of vision. After the stabilization process, there will be black lines on the top and bottom of the videos, so those have to be cropped on both videos so that they will line up when converted to a stereoscopic 3D video. We chose to use functions from OpenCV for this process because they have the necessary tools [3].

2.3.4 Correlating the frames of the two videos

The purpose of this process is to correlate the frames of the videos using the GPS time stamp data from the BLOB files. This will make it possible to convert the videos into one stereoscopic 3D video.

2.3.5 Converting two videos into one stereoscopic 3D video

The purpose of this process is to output the final video in the correct format to view a stereoscopic video in a VR headset. The software we chose to do this process is OpenCV [4].

2.4 Dependency Viewpoint

Component	Interconnection	Sharing	Parameterization
BLOB Parser	Input: GPS File From User	Output: Timestamp and GPS data in JSON Format	None
Correlate Frames	Input: JSON GPS and Timestamp File, User Uploaded Video Files	Output: Two Video Files Validated for Conversion	Files must be within 25 seconds in length.
Stabilizing Video	Input: Two Validated Video Files	Output: Two Video Files Stabilized With A Smoothing Radius of +- 30 Frames	None
Crop Videos	Input: Two Stabilized Video Files	Output: Two Video Files Without Stabilization Blank Spots	None
Convert to Stereoscopic File	Input: Two Stabilized and Cropped Video Files	Output: 3D Stereoscopic File in SBS format	None
Download Full or Sub-section of File	Input: Converted 3D Stereoscopic File	Output: (External Device) Full Converted file or Subset of Converted File	None

Dependency of processes on other processes.

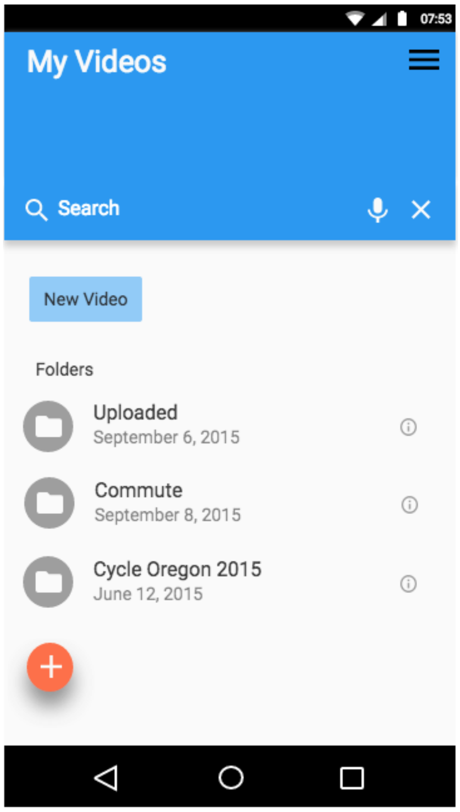
2.5 Patterns use Viewpoint

We will be utilizing the MVC software design pattern for the applications.

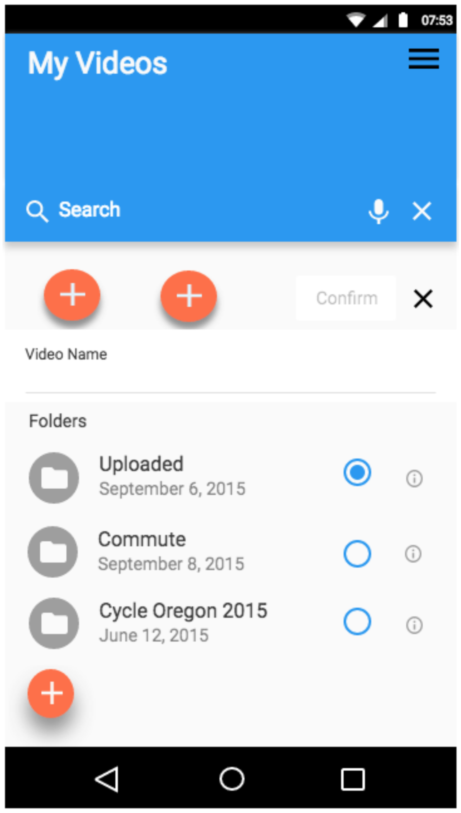
2.6 Interface Viewpoint

Use Case	Instructions
User upload files	Click on the upload files button -> Prompted to select 2 files for upload.
Select already processed files	Click on My Videos button and the page will navigate to the already processed video files.
Download processed files	On the My Videos screen select the download button once the files have been selected. A saving prompt will open.
Select subsection of processed file	Select the Download subsection button on the My Videos screen. The user will be prompted with inputs for the time sections that will be downloaded. The user will then click the select file to download the subsection of that file.

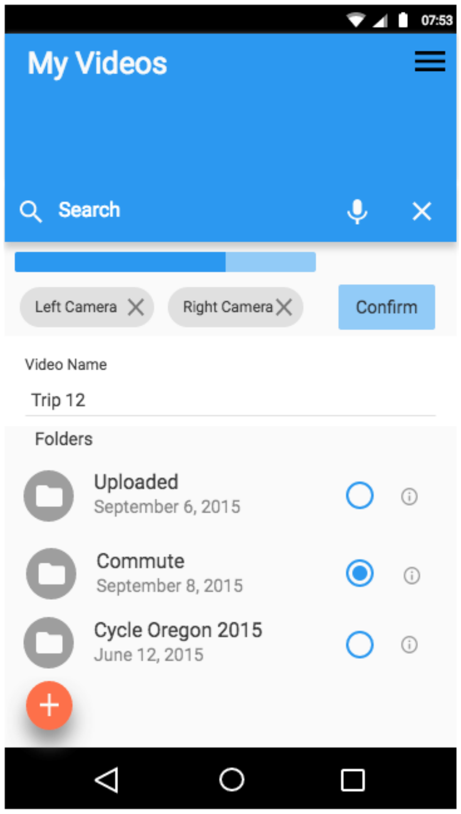
2.6.1 Process1: User selects video files to be processed:



2.6.1.1 User is at My Videos page [7]

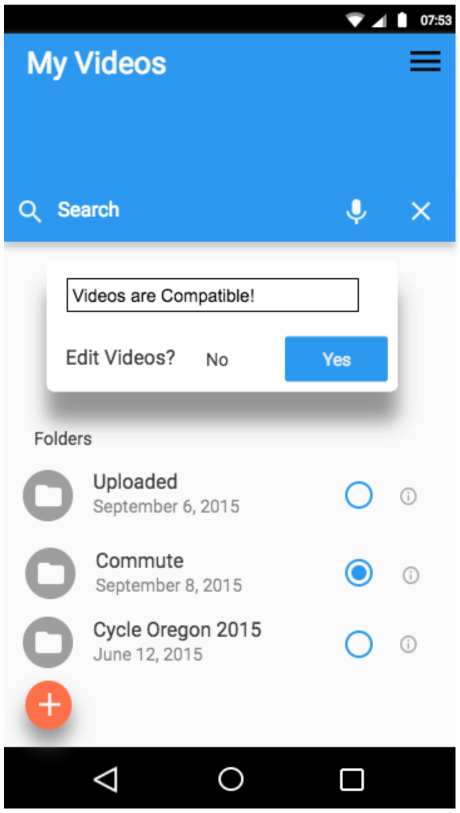


2.6.1.2 User clicks New Video button [7]

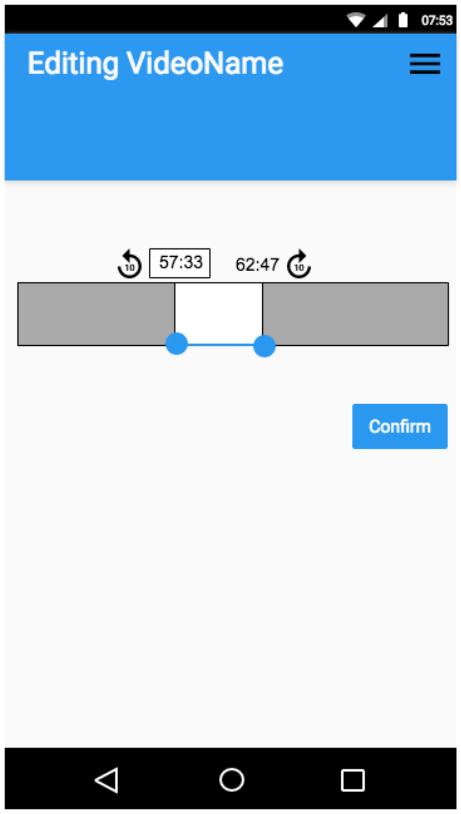


2.6.1.3 User provides two videos stored on devices file system, gives the video a name, and clicks the Confirm button [7]

2.6.2 Process2: User selects portion of video files to process:



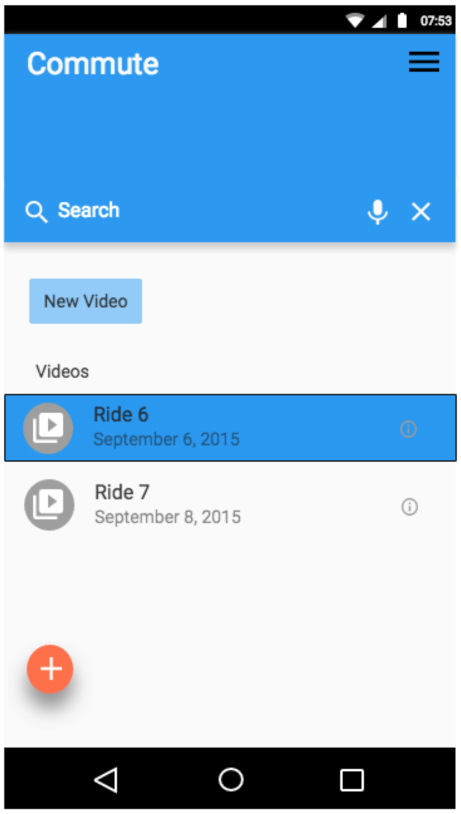
2.6.2.1 User clicks Yes to edit video [7]



2.6.2.2 Users makes use of video resizing tools to pick a subsection of video to process [7]

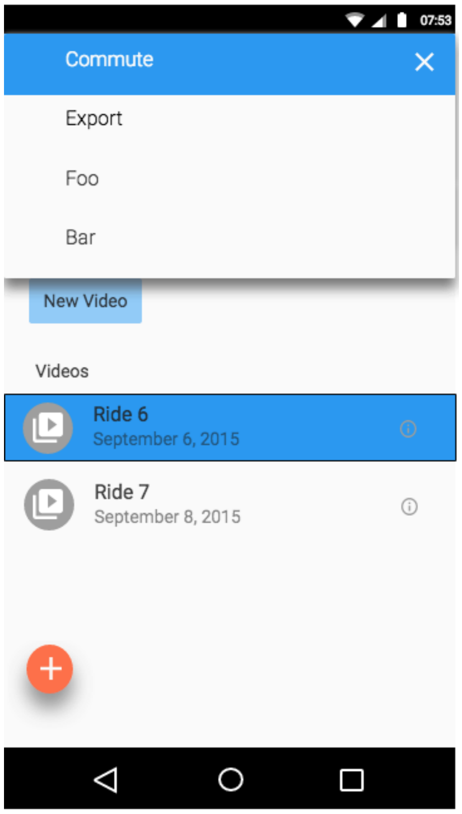
2.6.3 Process3: Select already processed files:

2.6.3.1 User is at My Videos page: See 2.6.1.1

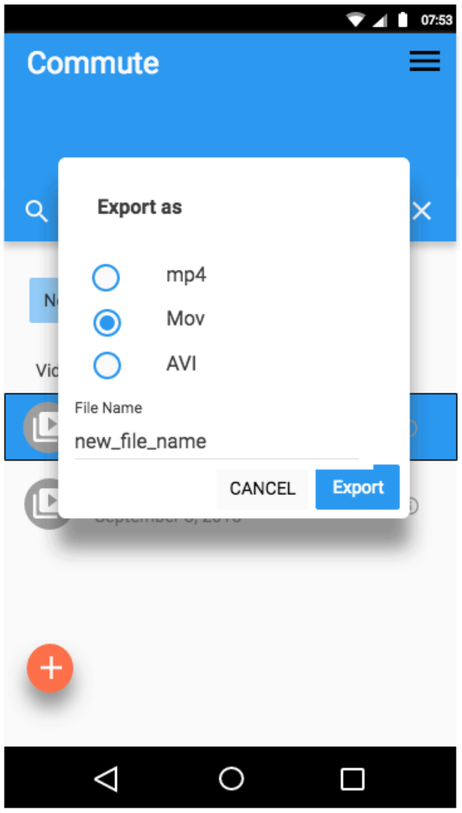


2.6.3.2 User navigates to Commute folder and selects desired video [7]

2.6.4 Process4: Export processed file:



2.6.4.1 User clicks menu expansion button and clicks Export option [7]

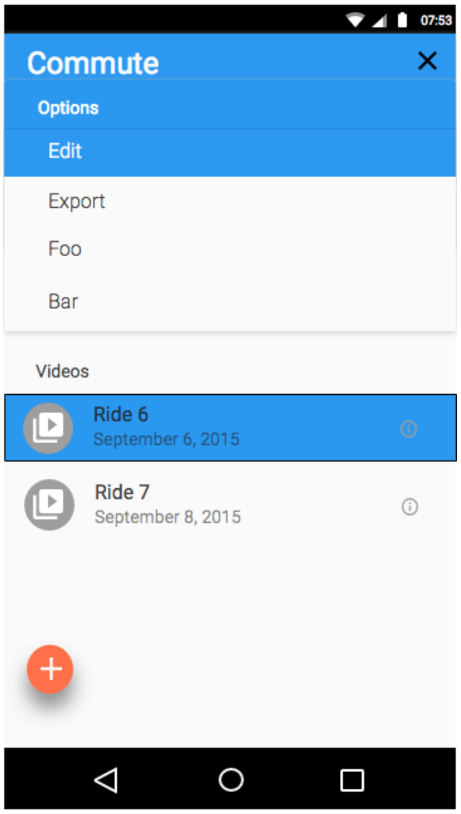


mp4 [✓ Mov] AVI

File Name
new_file_name

2.6.4.1 User chooses file format to export as and clicks Export button [7]

2.6.5 Process4: Select subsection of processed file:



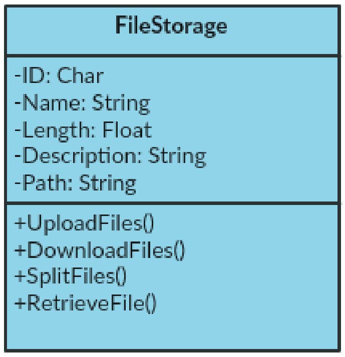
2.6.5.2 Users makes use of video resizing tools to pick a subsection of video to process [7]

2.6.5.1 User clicks menu expansion button and clicks Edit option: See 2.6.2.2

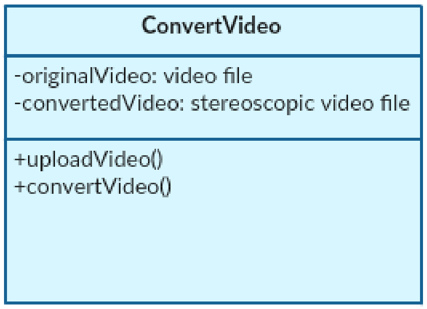
2.7 Structure Viewpoint

UML Class Diagrams for the different processes:

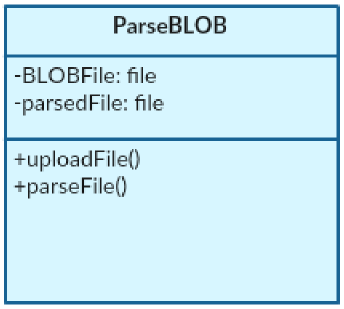
2.7.1 File Upload



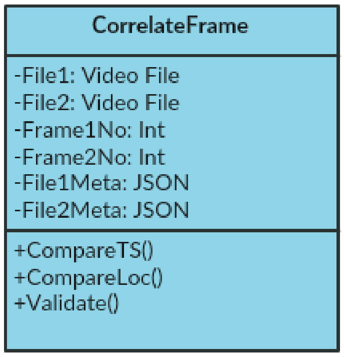
2.7.2 Converting Videos


ConvertVideo
-originalVideo: video file
-convertedVideo: stereoscopic video file
+uploadVideo()
+convertVideo()

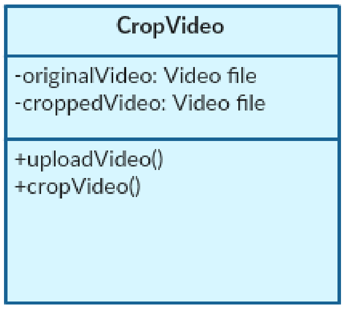
2.7.3 Parse BLOB



2.7.4 Correlate Frames



2.7.5 Crop Videos



2.8 Interaction Viewpoint

These are how the components of the system will be related and interact with each other.

2.8.1 Relationship 1: The output of the cameras is input for parsing the BLOB files and video stabilization.

2.8.2 Relationship 2: The output of video stabilization will be the input for cropping the videos.

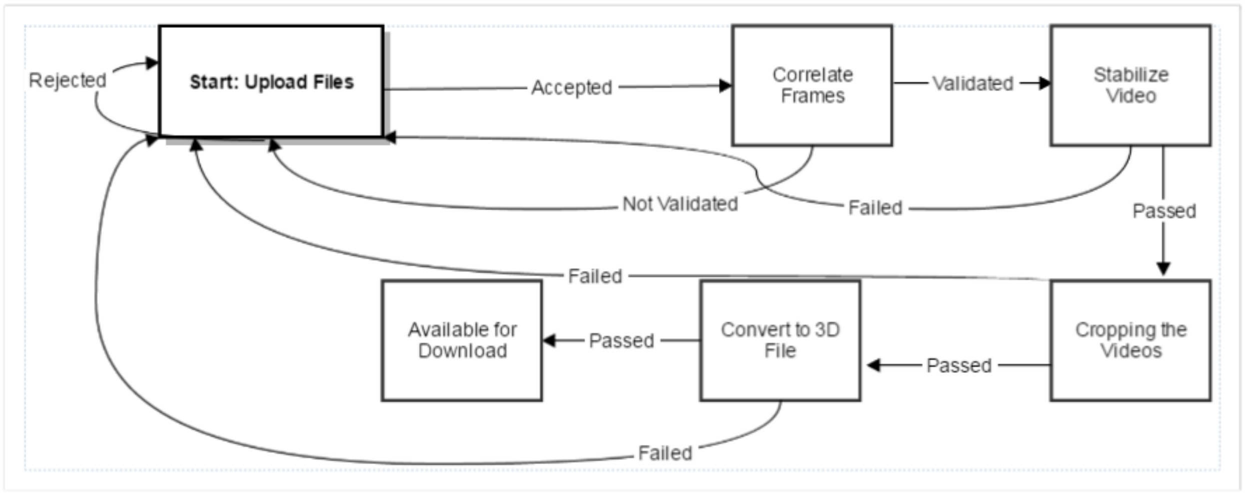
2.8.3 Relationship 3: The output of cropping the videos and output of parsing the BLOB files will be input for correlating frames.

2.8.4 Relationship 4: The output of correlating frames will be input for converting the two videos into one stereoscopic video.

2.8.5 Relationship 5: The output of converting the videos will be input for viewing the 3D video in a VR headset.

2.9 State Dynamics Viewpoint

A State Diagram depicting the flow of data through our system: [6]



2.10 Algorithm Viewpoint

2.10.1 Parsing the BLOB file:

1. This entity will make use of the metadata uploaded by a user of the system. If it is in a recognized format it will apply a known decoding algorithm and reformat the data to a specific format which will be used by the rest of the system.

2. The goal of this process is to reformat the metadata provided by the uploaded video, into a format that can be easily read by the rest of the program.

2.10.2 Stabilizing the videos:

1. This entity will most likely make use of a third party library to perform the video stabilization process. Our software will provide configuration data to this library. The configuration data will be based off set constants and the video file being processed.

2. The goal of this process is to reduce any shakiness in the video in the hope that the final product will be as stable as possible.

2.10.3 Cropping the videos:

1. This entity will most likely make use of a third party library to perform the video cropping. The decision of where and how much to crop the videos will be decided based on output from the video stabilization.

    a. OpenCV should provide pixel differences between the stabilized and unstabilized videos for each video file. Using this along with the timestamp and gps metadata we can find the largest non-outlier difference between the two files and crop the video at that point.

        i. Non-outlier: Find the average y pixel shift at each frame in each video, compare to frame in other video file at the same point, log and find average. Largest non-outlier difference will be the frame that is within 2 standard deviations from average.

2. The goal of this process is to remove vertical portions of the video that greatly differ from each other. These differing portions of video can be caused by the two cameras being set up at slightly differing angles or during the stabilization process.

2.10.4 Correlating the frames of the two videos:

1. The entity responsible for this process will make use of the metadata formatted by the BLOB parser. It will examine the timestamp data from each metadata file and find the first two frames that are nearest each other.

    a. Find video file which started latest. Store timestamp of the first frame captured by it, called start. This frame will be the first frame shown in the relevant side of the three dimensional video.

        i. E.g. if the right video file starts later than the left, start will be the first frame shown in the right side.

    b. Find frame in other video file which starts closest to start.

        i. Search metadata file for first frame that has a timestamp later than start, called post.

        ii. Find first frame before post, called pre.

        iii. Compare difference between start and post and start and pre.

        iv. Smallest timestamp difference found will decide which frame to treat as start of second camera.

2. The goal of this process is to remove vertical portions of the video that greatly differ from each other. These differing portions of video can be caused by the two cameras being set up at slightly differing angles or during the stabilization process.

2.10.5 Converting two videos into one stereoscopic 3D video:

1. This entity will most likely make use of a third party library to combine two processed videos into a single video file.

3 Conclusion

In this Software Design Document, we described how we are going to be developing our software, and how all of the components and processes work together. The main part of the system will be composed of several processes and the data flows through the system as follows: The user uploads two videos to the application and begins the conversion process. The videos get stabilized and cropped, then the frames are correlated, and then the two videos get converted into the stereoscopic format. The video is then exported so the user can view the video in a VR headset.