Animating Eyes

Gina B. Guerrero*
Digital Production Arts
Clemson University

Andrew Duchowski†
School Of Computing
Clemson University

Figure 1: *3D head model used for this study. The screenshots have sample data already keyframed.*

Abstract

Accurately and efficiently depicting the liveliness of eyes is not an easy task in animation. In this paper, we propose a method that utilizes eye tracking and motion capture technologies to animate 3D eyes. We measure it based on accuracy, efficiency, and user satisfaction.

CR Categories: I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—Animation;

Keywords: eye tracking, motion capture, animation

1 Introduction

In animation, eye movements are modelled through a series of aim constraints that are sometimes coupled with inverse kinematics (IK) [O'Hailey 2013; Vasconcelos 2011; Maraffi 2004]. These methods are limited in the sense that eye states may not be accurately portrayed, and the nature of eye jitter may be lost altogether, possibly producing a lifeless character.

In motion capture, electro-oculography (EOG) devices are used by placing markers on the face following a muscle-based method, peak movement method or a hybrid of both [Okun and Zwerman 2010]. However, data from these do not necessarily track eye movement, therefore eye tracking methods such as special contact lenses or a combination of video-oculography (VOG), an eye-image analysis using a camera and infrared oculography (IROG), a corneal-reflection of near infrared light source relative to the location of the pupil center, could be used in tandem [Kerlow 2001; Duchowski 2007]. The data derived from these methods provide a complete picture of eye motion that can be useful in animation.

As systems like Ergoneer's Dikablis, an infrared-based gaze tracking device, and Vicon's suite, a set of motion capture software, become compatible and integrated, the easier it is take eye tracking data and use it for animation purposes.

With that in mind, we want introduce a method to take such data from Dikablis and Vicon and use it directly with Autodesk's Maya. We believe this is a viable method of accurately portraying eye movement for animations without having to employ complicated constraints and IK or placing several marker detectors on the participant.

2 Background

For simplicity, we have broken down the different types of research into eye movement for the sake of recreating them in animation into the following models: data-driven, procedural, a hybrid of data-driven and procedural, and statistical. Since Fukuyama [2002] and colleagues have since demonstrated that gaze shifts can be enough to convey emotion and personality, most of the research do tend to focus on gaze.

Deng et al. [2005; 2003] take a data-driven approach which is the model our method falls under. However, in order to calculate the corresponding gaze for their animation, they manually estimate the eye direction via a frame by frame analysis of the training videos, which could result in some inaccuracy. Rhee et al. [2011] use an estimation method to determine the eye location by minimizing the sum of the pixel intensity.

Research done under the procedural model tend to focus on gaze shifts based on head movements and other parameters that are linked with neuroscience [Peters 2010; Andrist et al. 2012]. Another uses a sharing ratio based on view angle to determine composite head-eye movements along with body movements which are synchronized to a conversation state [Masuko and Hoshino 2007].

Lance and Marsella explore a hybrid of the two aforementioned models known as the the "expressive gaze model". In this model, they map physical behaviours with possible emotions that attribute to a gaze shift displaying those behaviours [2010].

*e-mail:gguerre@g.clemson.edu
†e-mail:duchowski@clemson.edu

Within the statistical approach, there has been research that use saccades in combination with empirical eye tracker data to model gaze while talking or listening during social interaction [Lee et al. 2002]. Incorporating Lee's idea of gaze behaviours and stochastic gaze shifts along with their studies of computer graphic animations, Queiroz at al. [2008] create a prototype for handling gaze in interactive environments.

A more recent study comparing simulated eye movement to no-eye tracking movement and real eye-tracking movement demonstrate that there is not a statistical difference between simulated and real-eye tracking when choosing between them. However, the simulated eye movement is limited to a specific scenario: when the participant is looking at their own virtual reflection [Borland et al. 2013].

Since most of the research mentioned above focus solely on gaze shifts with head/body movement or other behaviours, there is an aspect of eye motion that gets left out which is jitter. We want to include this motion as it is part of the natural eye movement. Most methods above also predict or simulate eye movement which may not always be an accurate representation for the scene at hand, as Queiroz et al. noted regarding Lee's statistical method [2007].

3 Methodology

3.1 System Architecture

3.1.1 Eye Tracker

The eye tracker used in this experiment is a modified version of Ergoneer's Dikablis to be binocular instead of monocular. It has an accuracy of less than 0.5 degrees at a sampling rate of 25Hz. One of the eye tracker software, D-Lab Recorder, runs on two Dell Latitude laptops with Windows 7, 32-bit, an i5 3320M at 2.60GHz, and 4GB of RAM: one for the right eye and one for the left. It is also a wearable eye tracker that has been further altered to already include the markers for motion capture.

3.1.2 Motion Capture

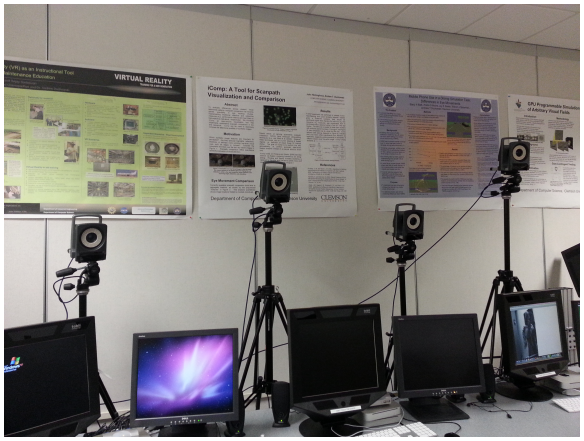


Figure 2: *Setup of the Vicon cameras in against the back wall.*

The motion capture system consists of four Vicon T40-S cameras set up in the back of the room. These have a 4.0 megapixel resolution, with a maximum frame rate of 515. However, it is set to run at a frame rate of 100. The software in use is Vicon Tracker 2.0 that has a native implementation of the Dikablis Eye Tracking system which allows gaze data to be included and has a validated latency of 2.5 milliseconds. This runs on a Windows XP Service Pack 3, 32-bit machine with an Intel Xeon 5130 at 2.00GHz and 3GB of RAM. D-Lab control, which is part of the Dikablis software suite and can simultaneously control the 2 D-Lab Recorders mentioned above, is run along side Vicon Tracker on this machine.

Tracker comes with a Virtual Reality Peripheral Network (VRPN) server and Vicon's own DataStream software development kit (SDK). This allows for easy integration with third party clients that can retrieve data from the server. The VRPN server is inherently a UDP protocol while DataStream is a TCP protocol, in which both are available when Tracker is live or replaying a recorded trial. Due to the fact that Vicon's DataStream has a method for calculating Z-Up to Y-Up axis and provides an easily accessible frame number for reference, a custom client was made to communicate with it instead of the VRPN server.

3.1.3 Custom 3rd Party Client

This custom client was written in C++ and employs the DataStream API to retrieve the following data: frame number, frame rate, head translation, head rotation (in Euler angles), and data from Dikablis for both right and left eye, as well as Vicon's estimated eye translation and gaze. There is also a boolean retrieved to tell if any of the data was occluded. The client formats the data into the following comma-delimited string (for readability purposes, newlines were inserted) and saves it to a local file:

```
[Frame Number],
[Frame Rate],
Head,
[Head Translation X],
[Head Translation Y],
[Head Translation Z],
[Head Translated Occluded],
[Head Rotation X],
[Head Rotation Y],
[Head Rotation Z],
[Head Rotation Occluded],
[Eye-R|EyeL],
[Eye Gaze X in 2D],
[Eye Gaze X in 2D Occluded],
[Eye Gaze Y in 2D],
[Eye Gaze Y in 2D Occluded],
[Eye Gaze X],[Eye Gaze X Occluded],
[Eye Gaze Y],[Eye Gaze Y Occluded],
[Eye Gaze Z],[Eye Gaze Z Occluded],
[EyePG-R|EyePG-L],
[Vicon Eye Translation X],
[Vicon Eye Translation Y],
[Vicon Eye Translation Z],
[Vicon Eye Translation Occluded],
[Vicon Eye Gaze X],
[Vicon Eye Gaze Y],
[Vicon Eye Gaze Z],
[Vicon Eye Gaze Occluded]
```

The client is a multi-threaded process that utilizes a template ring buffer queue to handle incoming data and writing out data. It is a single producer, single consumer model. It is also compiled and run from a laptop with Ubuntu 13.10 with 8GB of memory and an i5 M430 core at 2.27GHz, and with gcc verison 4.8.1 that is connected to the network.

3.1.4 Maya & Python Scripting

A python script was written in Maya 2013 to parse the flat file from the client and keyframe objects in the open scene named: Head,

EyeR, EyeL, EyelidR, and EyelidL. The following is the structure of the script:

```
blink = 0

with open(filename) as mydata:
  for line in mydata:
    parsed_line = parseline(line)
    frame_number = parsed_line[0]

  doheadrotate()

  try:
    i = parsed_line.index('Eye-R')
    if parsed_line[i+1]!='Eye-L':
      dounblink()
      blink = 0
      doeyerotate()
    else:
      blink = 1
  except:
    continue

  try:
    i = parsed_line.index('Eye-L')
    if parsed_line[i+1]!='EyePG':
      dounblink()
      doeyerotate()
  else:
    if blink == 1:
      doblink()
  except:
    continue
```

A blink is registered when the data was reported to be occluded for both right and left eyes, therefore a motion like winking would not be animated.

In order to calculate the angles for the eyes' rotations in the X and Y from Dikablis's GazeX, GazeY, and GazeZ, the following calculations were done within the Python script as part of *doeyerotate()*:

$$X_{rotation} = -\arctan(GazeX/GazeZ) \tag{1}$$

$$Y_{rotation} = \arctan(GazeY/GazeZ) \tag{2}$$

Due to the fact that the rotations, once calculated, were total rotations, there needed to be a safeguard to prevent the eyes from moving beyond the normal eye range. Without that, the resulting animation could be less than favourable. Based on Masuko's and Hoshino's [2007] assumptions regarding the average eye range, the Python script will also adjust the rotations for both head and eyes to fit the following model.

Due to mostly eye rotation:

$$0^\circ \leq |X_{rotation}| < 15^\circ \tag{3}$$

$$0^\circ \leq |Y_{rotation}| < 10^\circ \tag{4}$$

Due to head and eye rotations:

$$15^\circ \leq |X_{rotation}| < 30^\circ \tag{5}$$

$$10^\circ \leq |Y_{rotation}| < 20^\circ \tag{6}$$

Due to mostly head rotation:

$$30^\circ \leq |X_{rotation}| < 50^\circ \tag{7}$$

$$20^\circ < |Y_{rotation}| \tag{8}$$

Maya 2013 runs on a Windows 8, 64-bit machine that has 8GB of memory and an i7-3630QM core at 2.40GHz. Once the script completes, the animation can be played back at 25FPS to accommodate Dikablis's original sampling rate of 25Hz and recorded video which is at 25FPS.

3.2 Design

To keep the experiment from becoming too complex and diverting from the analysis of the proposed method, we decided to test this by simply focusing on a specific task. In this case: reading. Limiting the task to one with an easy to recognize eye pattern will help in qualitatively analysing the quality of the resulting animation. In addition, the resulting animation can also be compared to the recorded video from Dikablis.

Quantitatively, the data actually used to animate the model can be compared with the original set of data from Dikablis. A real occlusion, such as a blink, will appear in both the Dikablis data and Vicon data. An error in Vicon, such as the participant going out of range of the motion capture cameras, will only appear in Vicon. Any data collected that was calibrated with a high reprojection error (greater than or equal to 25) in Vicon's Tracker will be deemed inaccurate and unusable.

Applying a Savitzky-Golay smoothing filter [Gander and Hebek 2004] to both the Dikablis data and the Vicon data should help us determine when saccades occur. The comparison of the two can then also be used to determine how precise the animation is from the original data.

There will also be a post-survey for the participant to fill out. The survey is a combination of a 7 point Likert-scale that asks the following:

1.) How efficient do you think this process was?

2.) How accurate do you think the resulating animation is?

3.) How satisfied were you regarding the resulting animation?

And 2 free response questions that ask the participant regarding the reading material and for any comments they may have.

An incorrect answer to the reading material question will deem the data inaccurate, as we have no other way to determine that the participant actually read the stimuli.

3.3 Stimuli

The stimuli is nothing more than a one page paper with 22pt-size, Times New Roman font text regarding a random fact that has nothing to do with the study.

3.4 Participants

A sample of 16 college students from Clemson University participated in this study. Participants were primarily within the School of Computing department, specifically those with some experience in animation. Seeing as this is not quite a perception study on the animation itself, but rather the process that resulted in the animation, it was important to have participants with some background in this field.

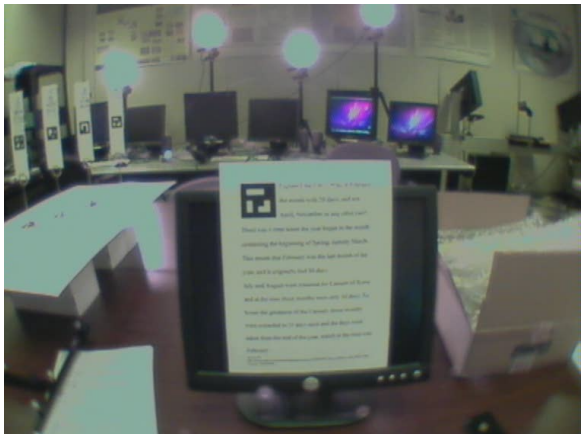


Figure 3: *Screen shot of the field camera view of the stimuli.*

3.5 Procedure

At the beginning of the study, we will inform the participant exactly what the study is for: using data from their eye tracking session to animate the eyes of a 3D model.

3.5.1 Calibrating Dikablis

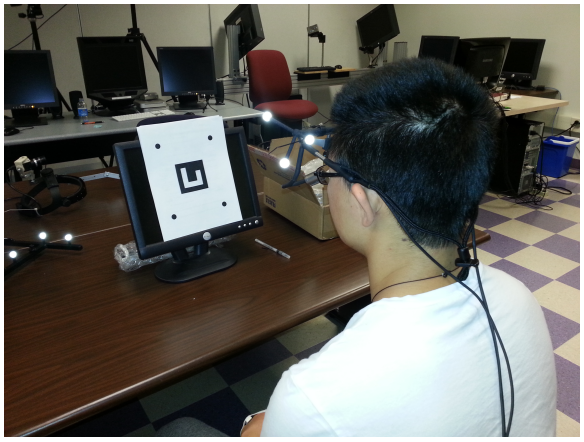


Figure 4: *Student with the eye tracker, facing calibration sheet.*

Calibrating Dikablis requires a few steps. The participant's eyes must be centered on the camera's views, and their pupil must be easily detected by Dikablis when moving within a set range.

Once the pupil has been detected, the user will be asked to look at a stand holding a sheet of paper with four dots, one at each corner, and an area of interest marker at the center. They will be asked to look at the bottom left, top left, top right, and then bottom right. Afterwards, to ensure calibration, we will ask them to follow a moving target with their eyes.

3.5.2 Calibrating Vicon

Calibrating Vicon with Dikablis will require the participant to hold a wand, and stare at the the middle, top marker. At first, they will be asked to hold it with their right hand, arm outstretched. They will then be asked to move it to the left, to the right, then to the center, while maintaining their gaze on the middle, top marker, and resisting the urge to move their head.

3.5.3 Recording & Playing Back Data

After calibration, the participant will be asked to read the stimuli. When they have completed the reading task, we will playback the recording in Vicon Tracker at half speed to ensure that the custom 3rd party client gathers enough frames from the DataStream server and saves them to a flat file.

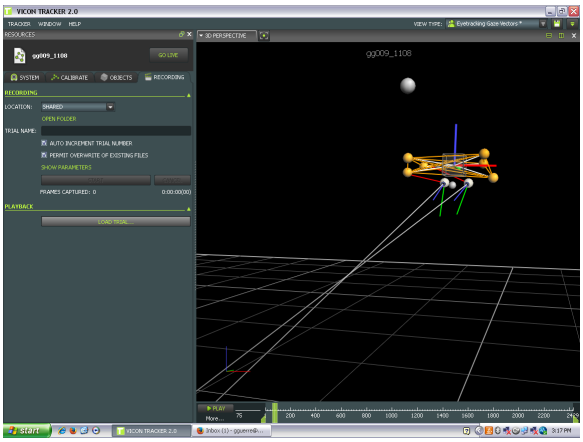


Figure 5: *Screen shot of Vicon Tracker in playback mode.*

The flat file can then be processed through the python script in Maya, and will automatically keyframe the correlating objects in the scene according to the recorded data.

The resulting animation will be played back to the participant (see **Figure 6**), and the post-survey will be given for them to fill out.

4 Results

4.1 General

Everyone answered the reading material question correctly, however, out of the 16 participants, 7 had known issues during calibration. Their data was not included in the following analysis with the exception of the survey results.

This does mean that out of all the data gathered, about 56% was usable without any post-processing in Maya (further explained in the Discussion section). Out of the 7, 2 of the data sets were completely unusable (eg, Dikablis crashed in the middle of the experiment, Vicon did not receive data for one eye), while 5 of the data sets were salvageable after some post-processing. Overall, this would mean that at least 86% of the data was usable.

4.2 Qualitative Analysis

4.2.1 Expected Motion Comparison

We chose a reading task as it would be easy to recognize eye motion engaged in reading. As noted by Olsson [2007], reading is generally a left to right, top to bottom motion within the western culture. However, because the animation model is facing the user, the left to right motion is reversed so it should be right to left (**Figure 6**).

Both left and right eyes were scored based on whether or not they appeared to follow that same pattern of motion. A score of 0 was given if it did not seem correct, otherwise it was given a 1.

Overall the captured eye data translated well into Maya in regards to the motion simulating the expected pattern mentioned above, with

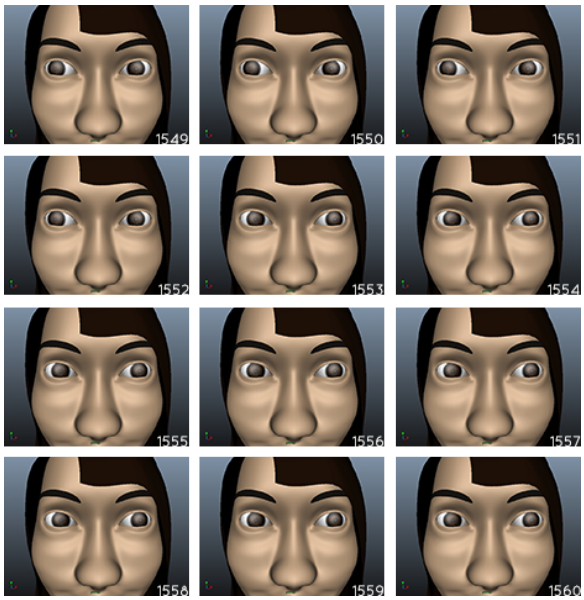


Figure 6: *12 frames already keyframed with the eye tracking data. Frames 1551-1554 indicate a saccade where the user has reached the end of the line and is moving to the start of the next line. This saccadic motion is reversed, therefore it is going from left to right. The following frames show a slower motion from right to left, indicating that the user has returned to a reading state.*

Participant	Left Eye	Right Eye
1	1	1
2	1	1
3	1	0
4	1	1
5	1	1
6	1	1
7	1	1
8	1	0
9	1	1

Table 1: *Results of the motion comparison.*

the right eye being the most problematic for 2 participants. With a little post-processing, this can also be corrected.

4.2.2 Animation-Video Comparison

We also overlayed the recorded videos from Dikablis on top of the resulting animations. The participant's pupil was estimated and aligned to the 3D model's estimated pupil. They were then both played back to determine if the model's estimated pupil generally followed the participant's estimated pupil (**Figure 7**). They were scored in a similar fashion as the expected motion comparison where 0 indicated that they did not at all seem to stay in sync, and 1 did seem to stay in sync.

The results for this comparison is identical to the expected motion comparison results table (**Table 1**).

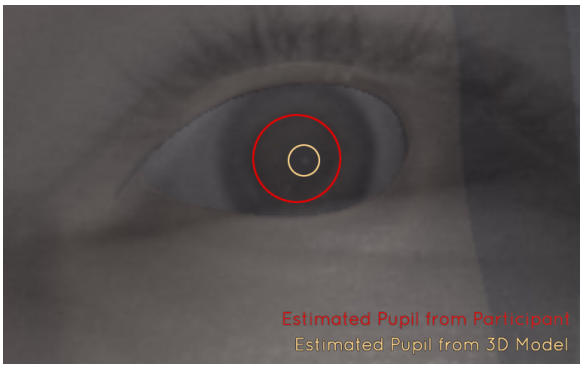


Figure 7: *A screen capture of the overlayed video, where the estimated pupils have been outlined.*

4.3 Quantitative Analysis

4.3.1 Data Smoothing

Pre-processing was done for both sets of data. Due to the fact that Dikablis would already be recording by the time Vicon starts to record, there was a need to trim the data to match the starting times with Vicon. While Vicon data was trimmed of any excess frames at the end, where there may be repeats from the beginning due to the playback looping in Vicon Tracker. This extraneous data was also used to fill in missing values in the beginning due to the 3rd party client not receiving them when the playback was just starting.

A 0th order interpolation was done on both Dikablis and Vicon data. This was simply filling in any missing values (eg, blinks) with the previous value.

Both sets of data were then smoothed using the Savitzky-Golay filter in NumPy [Haslwanter 2012]. This function is also capable of computing the derivative.

Dikablis used the following parameters $window\_size = 25, order = 2, deriv = 1, rate = 25$ due to its capture rate of 25Hz.

Whereas Vicon used the following parameters $window\_size = 101, order = 2, deriv = 1, rate = 100$ due to its capture rate of 100Hz.

Figures 8 and 9 are samples of the noise-reduced data and its derivative for both Dikablis and Vicon, separated by its horizontal and vertical components.

Visually, both sets of data look similar despite their different sample sizes.

4.3.2 Equivalence Test

To determine if the two sets of data per eye were statistically equivalent, two one-sided tests (TOST) were done on the velocity data. We chose to use the velocity instead of the position because of the difference in sampling rates. Therefore if the positions did differ significantly, as long as their velocities were similar, then it would mean that Vicon maintained a fairly similar speed and direction as what Dikablis recorded. This would also indicate that a loss in data integrity did not occur between the two systems.

In this case, our null hypotheses are:

$$H_0 a : \mu_1 - \mu_2 > \delta \tag{9}$$

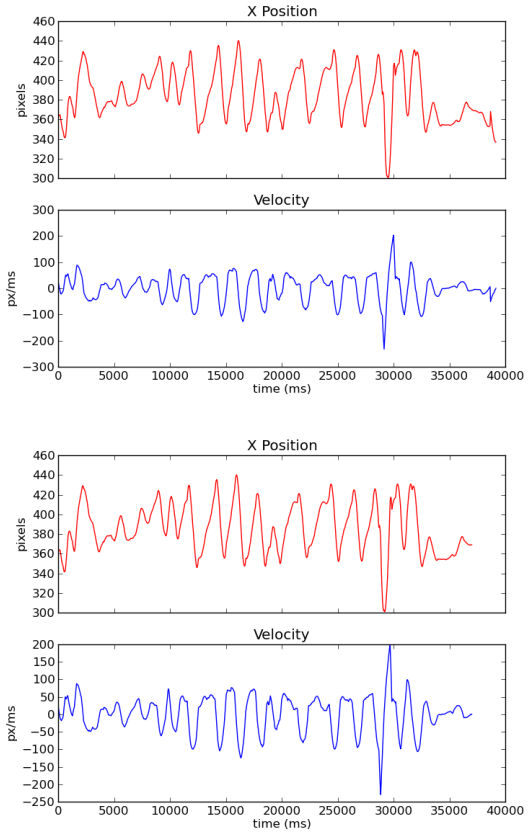


Figure 8: *A sample of data from Dikablis (top) and Vicon (bottom): smoothed position data and velocity in the horizontal component (x).*

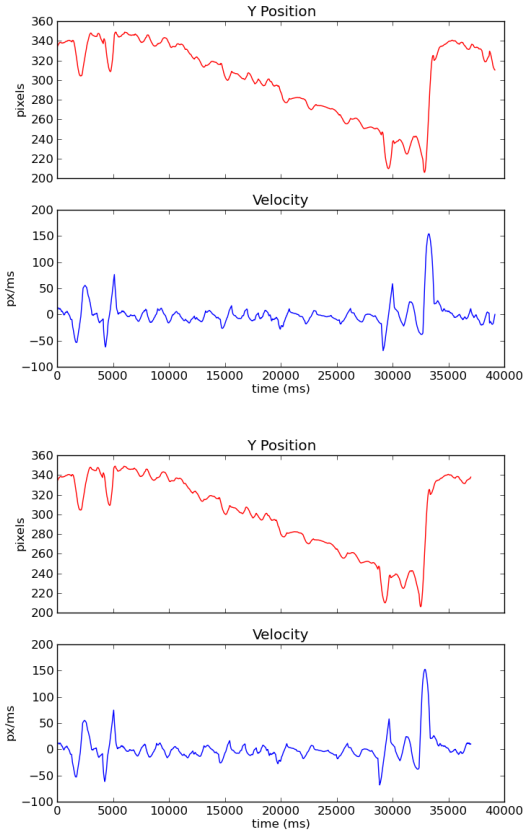


Figure 9: *A sample of data from Dikablis (top) and Vicon (bottom): smoothed position data and velocity in the vertical component (y).*

$$H_0 b^1 : \mu_1 - \mu_2 < -\delta \tag{10}$$

Where δ range is [-5, 5]. By rejecting these, we can conclude that our difference in data falls within the specified range.

We used Python's *statsmodels* module, which already has a function available for TOST: $ttost\_ind()$. The function was called with the following parameters: $low = -5, upp = 5, userver = unequal$.

Figures 10 and 11 are the results of the tests. The p-values highlighted in yellow were the greater of the two, and all of them reject the null hypotheses.

4.3.3 Occlusions

It was also important to ensure that Vicon did not add more occlusions than what Dikablis recorded. For simplicity, these occlusions were noted as blinks in the animation.

Both Vicon and Dikablis data were ran through a custom Python script that would look for sets of occlusions within the valid data and mark those sets as one occluded event.

Results (**Figure 12**) show that the amount of occluded events are about equal between the sets of data.

4.4 Survey Results

As mentioned earlier, all 16 participants' responses are included in this section. Although the actual survey had other questions, the three important ones were regarding the efficiency of the method, the accuracy of the animation, and their satisfaction with the animation.

It's interesting to note that when the responses were split between those whose data were analysed versus those whose data were not analysed, their opinion did not differ despite their differing experiences with the calibration process.

From the results (**Figure 13**), most participants felt that the method was somewhat efficient, the resulting animation was somewhat accurate, and they were somewhat satisfied with the animation.

When asked for other comments regarding their experience, their responses were a little bit more telling.

Comments from participants who had issues with the calibration process include: "The collected data could be more reliable," and "One eye's accurate, the other is not. It looks more accurate in Vicon than in Maya."

Comments from participants who had no issues with the calibration process include: "I believe this technique is a good starting point for eye animation..." and "It was very interesting how quickly my eye movement was tracked and then mapped to a model for animation."

Another noteworthy comment from a participant was regarding the

M(Dikablis)	SD(Dikablis)	M(Vicon)	SD(Vicon)	t(Lower)	p(Lower)	t(Upper)	p(Lower)	DoF
0.100933	30.8777	0.24021	29.92616	4.506461	3.55E-06	-4.76471	1.04E-06	1492.972
-1.98547	46.17037	-1.92043	45.64434	3.02558	0.001261	-3.10533	0.000967	1554.896
0.178565	54.77979	-0.03332	55.66667	2.920314	0.001769	-2.68287	0.003682	1882.906
0.710694	48.45224	-0.20178	48.26739	4.317877	8.19E-06	-2.98512	0.001431	2428.467
0.896606	39.30875	2.31268	28.42349	2.422992	0.007774	-4.33773	7.83E-06	1146.127
0.185295	22.6257	-0.05546	22.72453	6.154531	4.90E-10	-5.58906	1.37E-08	1402.862
-0.11711	31.87811	-0.14376	32.09429	4.180924	1.54E-05	-4.13659	1.87E-05	1400.358
0.433467	41.78821	0.660908	42.26365	3.174748	0.000765	-3.47734	0.00026	1547.08
0.743247	33.02909	0.426751	31.83002	5.025491	2.76E-07	-4.42715	5.07E-06	1773.965

M(Dikablis)	SD(Dikablis)	M(Vicon)	SD(Vicon)	t(Lower)	p(Lower)	t(Upper)	p(Lower)	DoF
0.205765	18.57364	0.24021	29.92616	7.707193	1.16E-14	-7.69405	1.29E-14	1497.173
2.956951	31.30765	-1.92043	45.64434	4.476354	4.07E-06	-4.53043	3.16E-06	1576.946
-0.57554	20.53881	-0.03332	55.66667	6.985994	1.96E-12	-7.98146	1.25E-15	1872.343
1.136194	22.53263	-0.20178	48.26739	9.355882	9.39E-21	-6.50813	4.64E-11	2338.972
0.487226	19.44175	2.31268	28.42349	10.72829	6.54E-26	-3.0829	0.00105	1102.773
0.37059	12.01364	-0.05546	22.72453	11.18368	3.64E-28	-10.9891	2.68E-27	1390.02
2.052879	14.22648	-0.14376	32.09429	9.360234	1.53E-20	-9.29192	2.81E-20	1396.357
-0.57532	23.41542	0.660908	42.26365	5.532575	1.85E-08	-6.33963	1.51E-10	1546.798
-0.11538	15.63175	0.426751	31.83002	9.496458	3.32E-21	-10.3941	6.45E-25	1801.513

Figure 10: *Results of TOST for the horizontal (top) and vertical (bottom) components for the right eye.*

M(Dikablis)	SD(Dikablis)	M(Vicon)	SD(Vicon)	t(Lower)	p(Lower)	t(Upper)	p(Lower)	DoF
0.101357	29.08217	0.139415	28.20374	4.889692	5.59E-07	-4.9647	3.83E-07	1499.569
-1.25057	39.92729	-1.26875	39.56087	3.561667	0.00019	-3.53587	0.000209	1565.915
0.388311	48.33891	0.003172	50.84182	3.396235	0.000348	-2.91045	0.001825	1939.242
0.812726	45.34379	-0.13609	44.71449	4.65528	1.71E-06	-3.17027	0.000771	2416.522
0.939191	52.90648	2.080112	32.80962	1.977628	0.024115	-3.14698	0.000848	1059.525
-0.1577	24.21672	0.384108	24.92029	4.855582	6.67E-07	-6.03578	1.01E-09	1418.618
-0.51707	34.06502	-0.53674	34.29144	3.908923	4.86E-05	-3.8783	5.50E-05	1402.363
-0.26515	51.73253	0.404235	52.23015	2.326924	0.010049	-3.04627	0.001178	1542.852
0.471347	33.73981	0.24449	32.78776	4.829557	7.43E-07	-4.41033	5.47E-06	1783.77

M(Dikablis)	SD(Dikablis)	M(Vicon)	SD(Vicon)	t(Lower)	p(Lower)	t(Upper)	p(Lower)	DoF
0.221392	16.58486	0.139415	28.20374	8.647305	6.63E-18	-8.61178	8.92E-18	1506.378
2.960967	28.44728	-1.26875	39.56087	4.965442	3.80E-07	-4.94888	4.13E-07	1593.519
-0.21441	13.29892	0.003172	50.84182	11.0032	1.37E-27	-12.4639	1.51E-34	1779.419
0.822989	17.00439	-0.13609	44.71449	11.14365	1.85E-28	-9.72693	2.91E-22	2415.235
0.23631	13.81564	2.080112	32.80962	16.35691	8.32E-54	-3.53522	0.000213	1006.068
0.463139	8.228404	0.384108	24.92029	17.01652	2.16E-59	-15.2871	3.72E-49	1378.967
2.390652	14.94039	-0.53674	34.29144	8.803736	1.95E-18	-9.03898	2.61E-19	1377.38
-0.4438	24.31411	0.404235	52.23015	5.072848	2.20E-07	-6.35959	1.33E-10	1542.842
0.047787	14.3299	0.24449	32.78776	10.22575	3.32E-24	-11.4114	1.75E-29	1820.283

Figure 11: *Results of TOST for the horizontal (top) and vertical (bottom) components for the left eye.*

use of all the data: "It might be good to do for realistic movies, but for animation some of the detailed data might not be necessary."

5 Discussion

5.1 Discrepancies

Relating to the survey results: participants did not see their animations with the complete and interpolated data. This led to some very noisy and inaccurate eye motions in their animations. If given a chance to show them what their animation looks like with the cleaned up data, their responses may differ.

However, since this study was primarily about the method, the survey responses from this trial was to serve as motivation for future work on perception of real eye motion within animation.

5.2 Post-Processing

Earlier, we mentioned a post-processing step in Maya. For animations that had eyes in an awkward position (**Figure 14**) yet seemed to have correct motion, it is possible to create a new grouping for that eye in Maya (shortcut: Ctrl+G). This group then allows for the eye to be rotated to a more fitting position and still maintain the motions that were already keyframed from the eye tracking data.

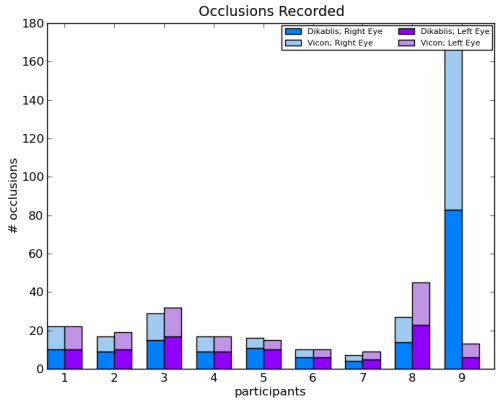


Figure 12: *Occlusions recorded in Dikablis and Vicon.*

5.3 Accuracy

As indicated by the quantitative analysis, the data remains fairly precise between Dikablis and Vicon.

Visually, the resulting animations look acceptable.

Unfortunately there is still the fact that 7 out of 16 data sets were not viable for analysis due to known calibration or other miscellaneous errors. Of those 7, 5 data sets were salvageable with the post-processing step mentioned above. This still leaves 2 unusable data sets.

Based on survey results, most users felt that the accuracy of their animations were somewhat accurate with some comments about how the data could be more reliable and accurate.

5.4 Efficiency

Even with the calibration errors, no participant stayed longer than the allotted 20 to 30 minutes. Within that time, they went through the calibration, read the stimulus, watched the playback from Vicon, watched their resulting animation in Maya, and completed their survey.

Once the data was collected, we were able to replay and reuse it on different models in Maya as long as the model has objects appropriately named as mentioned in Section 3.1.4 (**Figure 15**). Having this functionality also made the analysis very easy.

Based on survey results, most users felt that the process was somewhat efficient with comments noting how quickly their eye movements were tracked and applied to a 3D model for animation.

5.5 Satisfaction

In terms of satisfaction, most participants felt somewhat satisfied with their resulting animation.

6 Future Work

In regards to the method, there is a need test the implemented process of including head rotations based off the eye rotations in addition to the rotations given by Vicon. A task involving rolling one's neck may be better suited for this than reading.

The method also needs to be tested with tasks outside of reading.

Enhancements to the model and method, such as more realistic eye lids and their motions when the eye is rotated a certain way since eye lids do not stay stationary as one looks up and down.

Testing the method in real-time could also be done. It was written originally with the hopes of being used in real-time, but time constraints limited it to offline use.

Lastly, there is still the question of perception regarding eye motion: would one be able to differentiate between real jitter from eye tracking data versus a simulated jitter?

7 Conclusion

Despite the fact that this method can be bogged down with calibration difficulties, the eye tracking data can be very useful for an animator. Once the data has been recorded, it can be reused several times. Simple errors such as positioning can be easily fixed. This method also allows for an efficient way to conduct future tests regarding real eye motion in animation.

Acknowledgements

We would like to thank the students in Clemson University's Digital Production Arts program who took time out of their endlessly busy schedules to participate.

References

ANDRIST, S., PEJSA, T., MUTLU, B., AND GLEICHER, M. 2012. A head-eye coordination model for animating gaze shifts of virtual characters. In *Proceedings of the 4th Workshop on Eye Gaze in Intelligent Human Machine Interaction*, ACM, New York, NY, USA, Gaze-In '12, 4:1–4:6.

BORLAND, D., PECK, T., AND SLATER, M. 2013. An evaluation of self-avatar eye movement for virtual embodiment. *Visualization and Computer Graphics, IEEE Transactions on 19*, 4, 591–596.

DENG, Z., LEWIS, J. P., AND NEUMANN, U. 2003. Practical eye movement model using texture synthesis. In *ACM SIGGRAPH 2003 Sketches & Applications*, ACM, New York, NY, USA, SIGGRAPH '03, 1–1.

DENG, Z., LEWIS, J., AND NEUMANN, U. 2005. Automated eye motion using texture synthesis. *Computer Graphics and Applications, IEEE 25*, 2, 24–30.

DUCHOWSKI, A. 2007. *Eye Tracking Methodology: Theory and Practice*. Springer.

FUKAYAMA, A., OHNO, T., MUKAWA, N., SAWAKI, M., AND HAGITA, N. 2002. Messages embedded in gaze of interface agents — impression management with agent's gaze. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, ACM, New York, NY, USA, CHI '02, 41–48.

GANDER, W., AND HEBEK, J., Eds. 2004. *Solving Problems in Scientific Computing Using Maple and Matlab*. Springer.

HASLWANTER, T., 2012. Scipy cookbook: Savitzky golay filtering.

KERLOW, I. V. 2001. *The Art of 3D: Computer Animation and Effects*. John Wiley & Sons.

LANCE, B., AND MARSELLA, S. 2010. The expressive gaze model: Using gaze to express emotion. *Computer Graphics and Applications, IEEE 30*, 4, 62–73.

LEE, S. P., BADLER, J. B., AND BADLER, N. I. 2002. Eyes alive. In *Proceedings of the 29th annual conference on Computer graphics and interactive techniques*, ACM, New York, NY, USA, SIGGRAPH '02, 637–644.

MARAFFI, C. 2004. *Maya Character Creation, Modeling and Animation Controls*. New Riders.

MASUKO, S., AND HOSHINO, J. 2007. Head-eye animation corresponding to a conversation for cg characters. *Computer Graphics Forum 26*, 3, 303–312.

O'HAILEY, T. 2013. *Rig it Right! Maya Animation Rigging Concepts*. Taylor & Francis.

OKUN, J. A., AND ZWERMAN, S., Eds. 2010. *The VES Handbook of Visual Effects*. Elsevier Inc.

OLSSON, P., 2007. Real-time and offline filters for eye tracking.

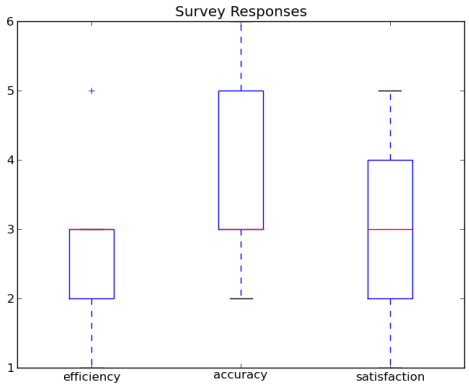
PETERS, C. 2010. Animating gaze shifts for virtual characters based on head movement propensity. In *Proceedings of the 2010 Second International Conference on Games and Virtual Worlds for Serious Applications*, IEEE Computer Society, Washington, DC, USA, VS-GAMES '10, 11–18.

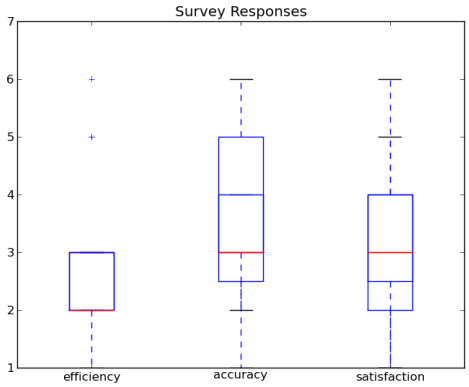
QUEIROZ, R. B., BARROS, L. M., AND MUSSE, S. R. 2008. Providing expressive gaze to virtual animated characters in interactive applications. *Comput. Entertain. 6*, 3 (Nov.), 41:1–41:23.

RHEE, T., HWANG, Y., KIM, J. D., AND KIM, C. 2011. Real-time facial animation from live video tracking. In *Proceedings of the 2011 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, ACM, New York, NY, USA, SCA '11, 215–224.

RODRIGUES, P., QUEIROZ, R., BARRO, L., FEIJ, B., VELHO, L., AND MUSSE, S. R. 2007. Automatically generating eye motion in virtual agents. In *Proceedings of the IX Symposium on Virtual and Augmented Reality*, SVR2007, 84–91.

VASCONCELOS, V. 2011. *Blender 2.5 Character Animation Cookbook*. Packt.





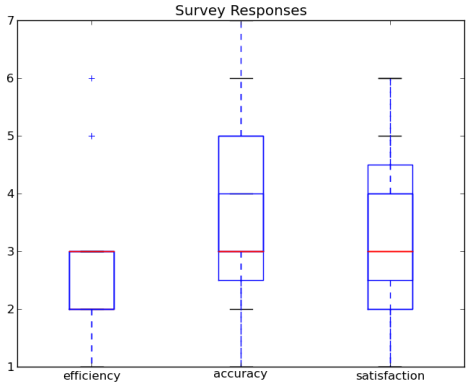


Figure 13: *Survey results of those whose data was used in the analysis portion (top), those whose data was not used (middle), and the combined results (bottom). A score of 1 indicates that it was very efficient/accurate/satisfied while a score of 7 indicates that it was very inefficient/inaccurate/unsatisfied.*

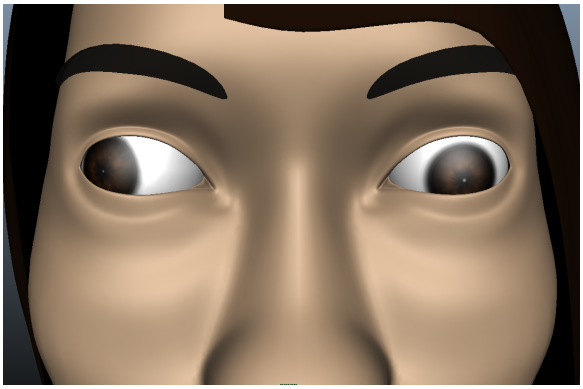


Figure 14: *An example of awkward positioning that would require the post-processing step.*

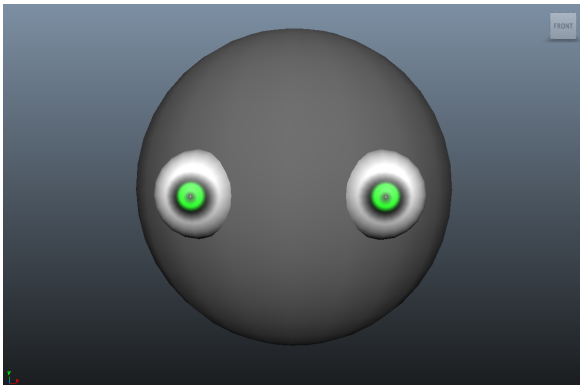


Figure 15: *This is the minimalistic model that was used while testing the method. Although this model did not have any objects for eyelids, data for eye and head rotations were still usable with this model.*