Cursor Control with Webcam-Based Head Pose Estimation

Cindy Jiang
cindyj@stanford.edu

Ethan Li
ethanli@stanford.org

Vinson Luo
vluo@stanford.org

Abstract

Head pose estimation is a fundamental computer vision problem that has many important applications, such as face recognition, head gesture recognition, or real-time attention assessment.

In this paper, we use a stereo webcam system to build an intuitive real-time control system that moves a cursor to the location a user's head is pointing. Given input from two webcams, we calibrate a rigid model of the head by triangulating corresponding facial landmarks from both images. From these 3D reconstructions of the head, we calculate the head pose of the user, using RANSAC to find inliers and optimizing with SVD-based partial Procrustes superimposition. We use this estimated head pose to project a cursor onto the screen that follows the user's head direction in real-time. We employ mean filtering and Kalman filtering to reduce noise in two parts of our pipeline—though such filtering decreases the responsiveness of our system, it is essential to attaining the level of precision desired in a cursor control system. Our final system allows users to control a cursor with a precision of around 10 pixels at most screen locations while remaining physically intuitive to control.

1. Introduction

Head pose estimation is the task of calculating the orientation and position of a head in the world, given a set of 2D and/or 3D images. Our specific problem is stated as follows: given real-time webcam footage of a user from two webcams, move a cursor to the position that the user's head is pointing towards.

The algorithm that we are implementing is built upon the gazr library created by Lemaignan *et al*. [10], which implements 3D head pose estimation using the dlib face detector library [7] and OpenCV's solvePnP function [1]. We compare the monocular webcam approach of gazr with our implementation of a stereo webcam approach. We hypothesized that the stereo webcam configuration better approximates the depth of the user's head, which provides improved accuracy when determining the desired position of the cursor on the screen.

To create a rigid model of the head given a set of webcam images, we start by using the dlib [7] facial landmark detector to extract facial landmarks from each image. Corresponding landmarks are then triangulated using a nonlinear method to optimize reprojection error. From that point, we observe real-time images from the webcams and use the Kabsch algorithm to compute the transformation matrix between the observed head pose and initial calibrated head pose. RANSAC is used with a 2 cm threshold in order to eliminate outliers. In addition, we reduce input noise by applying a sliding window mean filter on facial landmark points and a Kalman filter on the position of the on-screen cursor.

2. Related Work

There are a number of previous works that are relevant or similar to the problem we are addressing. Murphy-Chutorian *et al*. [13] perform a comprehensive review of a variety of head pose estimation methods, including appearance template methods, detector array methods, nonlinear regression methods, manifold embedding methods, flexible models, geometric methods, tracking methods, and hybrid methods. They also propose methods to obtain a ground truth estimate for evaluation, which ranges from subjective methods (e.g. manually assigning head poses to images) to more rigorous methods (e.g. magnetic sensors or optical motion capture systems with near-infrared cameras).

Hannuksela [3] implements cursor control from head pose estimating using a monocular camera system composed of an initialization stage and a tracking stage as well, but considers facial feature tracking and gaze detection in addition to head pose estimation. Their pose estimation system applies a Kalman filter, which is an optimal nonlinear estimation technique that can be used to generate a 3-dimensional motion estimation of a rigid model of the head.

Other researchers use a stereo camera configuration approach, only a couple of which are included in this section. Jiménez *et al*. [5] employ a method similar to the method above. First, they construct a rigid model of the head using thirty feature points identified by the Harris algorithm. The SMAT algorithm is used to track the two images independently, after which incorrectly tracked points are elim-

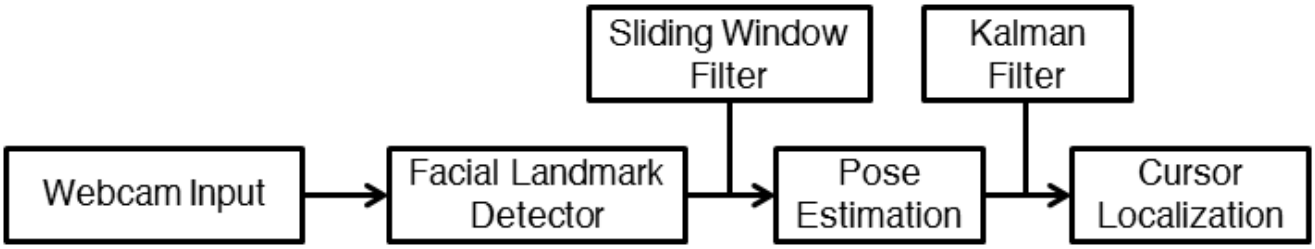

Figure 1. Our overall pipeline for webcam-based cursor control..

inated through RANSAC and the head pose is calculated with POSIT. Gurbuz *et al*. [2] implement a stereovision method that does not use a rigid model or initialization; they use the eye positions to reconstruct the 3-dimensional face, calculate a face plane estimation, and obtain the head pose.

Head pose estimation is a tool that has many applications in computer vision. It can be used to address the problem of eye gaze tracking, which has applications in human-computer interaction, human behavior analysis, and assistive driving systems. For example, head pose indicators can be used as technological cues, such as to a Wii system [15]. In addition, due to occlusion or ambiguity when detecting eyes in input images, head pose estimation can help define a persons field of view [16]. There is a correlation between attention and eye gaze or head pose direction [9]. This connection can be used in assistive driving systems to track the drivers attentional focus, which should be on the road [12].

In the taxonomy of approaches proposed by Murphy-Chutorian *et al*. [13], we implement a model-based geometric method using the estimated three-dimensional locations of detected facial landmark features. We also apply Kalman filtering and eliminate facial landmark outliers with RANSAC.

3. Methods

We compare two different methods for cursor control: control using a Perspective-n-Point based monocular camera baseline approach and control using a triangulation based stereo camera setup. Our overall pipeline for cursor control is shown in Figure 1.

3.1. Webcam Input

We use the video streams from one to two calibrated webcams as the input into our pipeline, as seen in Figures 2 and 3. For both the monocular and stereo approaches, the calibration of the cameras must be accurate for sensible results. We use the StereoVision library [8] with a printed chessboard to calibrate both cameras. For the stereo approach, the rotation and translation between the two cameras must also be known. In most stereo webcam configurations, the cameras can be assumed to be parallel, so only the distance between the webcams is required. In addition, the webcams should be close enough to each other that all the facial landmarks should be visible from both views when the user's head is at a distance at which their face is detected.

Figure 2. Stereovision configuration. Webcams are mounted on top of the monitor which displays the cursor.

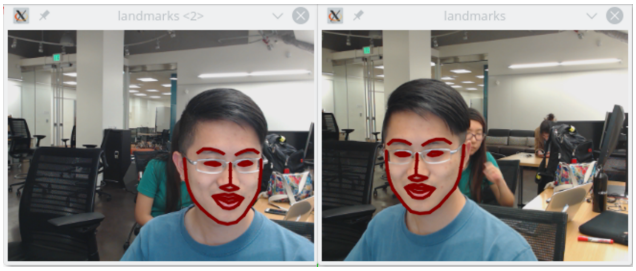

Figure 3. The two webcam views from the stereovision configuration. The dlib facial features are outlined in red.

3.2. Facial Landmark Detector

Given one or two simultaneous frames from our webcam input, we then apply a facial landmark detector to recover the 2-dimensional locations of keypoints on the user's head. For our system, we use the facial landmark detector provided by the dlib library [7] to extract the 68 different facial features pictured in Figure 4, which are outlined on the webcam images as pictured in Figure 3.

An important consideration is that not all of the detected facial landmarks are rigidly connected. The points corresponding to the lower chin, mouth, and eyebrows can move significantly relative to other points even when a user's head pose remains the same, particularly when a user is talking.

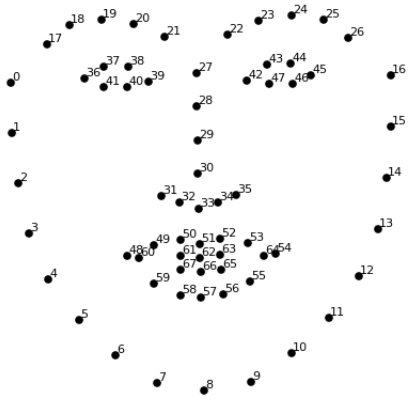

Figure 4. The 68 facial features returned by the dlib facial landmark detector.

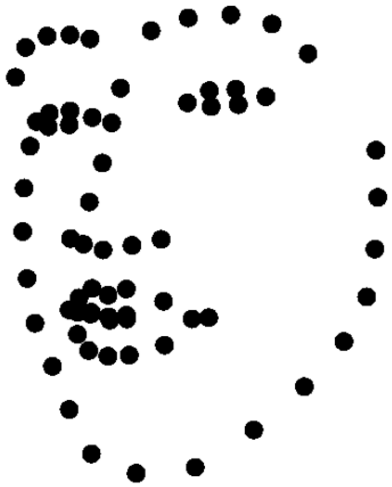

Figure 5. 3d point cloud of facial landmarks after triangulation.

3.3. Pose Estimation

We compare two different methods of estimating head pose for cursor control: the monocular perspective-n-point based approach and the stereo rotation matching approach. Both approaches are alike in that they require a calibration stage to establish an initial head pose relative to which all later poses are computed.

The calibration stage corresponds an initial pose of the user's head to a position on the screen. During this stage, the user holds their head still while looking at a specified point on the screen. The user signals for the calibration phase to end when a calibration visualization seems stable, indicating that a good initial model has been established. The webcam images at that moment are captured and used to build the initial head model.

3.3.1 Monocular Approach

In the monocular approach, the facial keypoints corresponding to eight relatively rigid points on the face are compared against a fixed 3-D model of a generic user's face, obtained from [17]. Finding the rotation and translation that reproject the fixed 3-D model onto the observed 2-D pixel locations reduces to solving the perspective-n-point problem with two cameras. We use the rotation and translation minimizing the mean squared error between the reconstructed 3d points and original model.

3.3.2 Stereo Approach

In the stereo approach, facial landmarks are first recovered independently from each image. We then triangulate the pairs of matching facial landmarks to recover their approximate 3-D locations. This is accomplished by first setting an estimate for the 3-D location that minimizes the least squares distance from both triangulating lines. We further refine this estimate using a nonlinear iterative method that minimizes the reprojection error from the computed 3-D point to the two observed 2-D image points.

Triangulating all pairs of facial landmarks produces a 3-D point cloud of facial landmarks, as shown in Figure 5. For the stereo approach, we set the reference model of the system to be the point cloud triangulated when the user is looking at the center of the screen during head pose calibration. Once this reference model is set, we fit the optimal rotation and translation that transform the reference model into the currently observed 3-D model, minimizing the average squared distance between matching points. We find the optimal R and T using the SVD-based partial Procrustes superimposition algorithm [6].

3.3.3 RANSAC-Based Outlier Removal

Because some facial landmarks can be localized very poorly by the landmark detector (leading to poor triangulation) we enhanced the partial Procrustes superimposition algorithm with a RANSAC based approach that filters out outliers at every time step. In our algorithm, we first iteratively chose random sets of 4 facial landmarks in the 3d point cloud, each of which uniquely describes a transformation consisting of rotation and translation. For each obtained rotation and translation, we determine the inliers for the particular transformation as the points that contain a low difference between their reconstructed and observed locations. We found that in practice a 2 centimeter threshold for determining inliers worked well at improving the stability of our system.

Only the points in the largest inlier set obtained through this iterative process, which we run for 50 iterations, are

then used to reconstruct R and T using the partial Procrustes superimposition algorithm.

3.4. Cursor Localization

Localizing the cursor directly in front of the user's head given the rotation and translation from the reference position simply the task of finding the intersection between the ray emanating from the user's head in their head pose direction and the screen plane. In order to map this intersection (measured in the camera's reference system) to a pixel value, several additional pieces of information are needed:

1. We assume that the webcam(s) lies in the screen plane and points normal to the screen plane. This is usually the case for most webcam setups.
2. We need to know the resolution of the monitor to determine the conversion between pixels and centimeters.
3. We need to know the displacement from the webcam to a known pixel location on the screen.
4. We need an initial correspondence mapping a head pose to a particular pixel location (obtained during the calibration phase).

With these pieces of information, the cursor location corresponding to a new head pose can be computed exactly. Even if the computed cursor location is not perfectly in front of the user, the feedback loop provided by the constant movement of the cursor in response to head motion provides an intuitive interface to accurately control cursor movements.

3.5. Noise Filtering

As mentioned earlier, the dlib facial landmark detector is rather prone to noise. Triangulation of the landmarks is also very sensitive to input noise, so the final cursor location is unstable. We want to make the cursor path smoother and to minimize cursor instability when the user's head is held stationary. To accomplish this, we use noise reduction methods. There are two instances in our approach where we apply noise filtering algorithms: a sliding window mean filter on the extracted facial landmark points and a Kalman filter on the location of the cursor.

After obtaining the facial landmarks, we apply an independent sliding window mean filter of size 20 to the position of each landmark of each image. This attenuates Gaussian noise in the location of 2-D facial landmark positions returned by the dlib facial landmark detector when the user's head is still. While using such a linear filter has the potential to deform the triangulated keypoints when the user performs rapid rotation of the head, the additional stability provided during finer movement is essential to allowing everyday cursor operation.

We also apply a Kalman filter which estimates cursor location, velocity, and acceleration from cursor positions computed during the cursor localization step to smooth the on-screen trajectory of the cursor. We find that this filtering provides further stability in cursor control, albeit at the cost of some responsiveness. This form of Kalman filtering, based on the physical laws of motion, has the advantage of being intuitive, allowing for fast user localization of the cursor to a desired point. For improved cursor positioning stability, we used the velocity and acceleration estimates from the Kalman filter to lock the cursor position whenever the the cursor became stationary; hysteresis was added to the transitions between the stationary and non-stationary state.

4. Evaluation

When we evaluated the baseline monocular system, we were unable to achieve accurate estimates of the depth (z-position) of the user's head for our cursor control system. As a result, we focus on evaluating the performance of our stereovision-based system.

Because the task of cursor control is a user-centric one, validation of the success of our stereovision-based system must be evaluated through subjective tests. Quantitative evaluation of the accuracy and stability of head pose estimation can be done before user validation. Robust measurement of accuracy, as described by [13], requires use of precise head-mounted inertial sensors or optical motion capture systems. We did not have access to such equipment, and the cursor positioning task can tolerate minor inaccuracies because the user can make compensatory movements to reach their desired cursor position. Thus, we chose to evaluate accuracy qualitatively while evaluating robustness to noise quantitatively.

4.1. Qualitative Results

During qualitative evaluation of our system (refer to supplement for a demonstration video), we found that our system generally accomplished the cursor control task, but with accuracy and stability of the cursor position were limited. The system responded well to coarse movements such as scanning the cursor across the screen, but fine head movements sometimes led to undesirable results such as overshooting or undershooting. When the user's head was pointed towards the edges of the screen, the cursor sometimes would converge on a very inaccurate position due to incorrect estimation of facial landmark positions by dlib's detector. Additionally, the cursor's position would still fluctuate around its target position due to noise which passed through our noise filters; this behavior was particularly noticeable when the user's head remained still. The instability of the cursor position reduced the ease of precisely adjusting the cursor's position through small head movements.

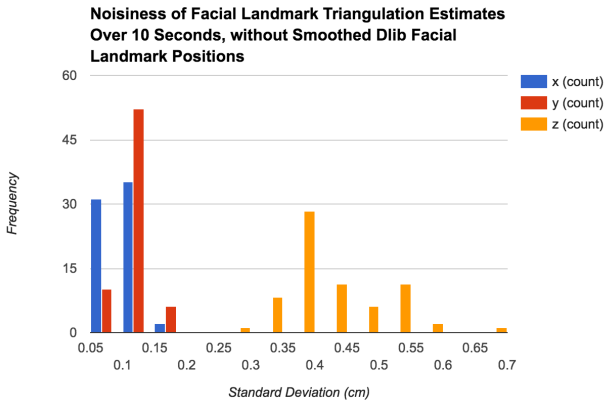

Figure 6. Standard deviations of the triangulated facial landmark position estimates over 10 seconds, without noise filtering.

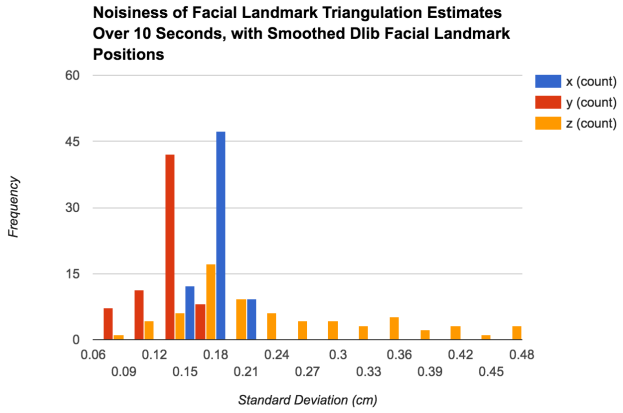

Figure 7. Standard deviations of the triangulated facial landmark position estimates over 10 seconds, with noise filtering.

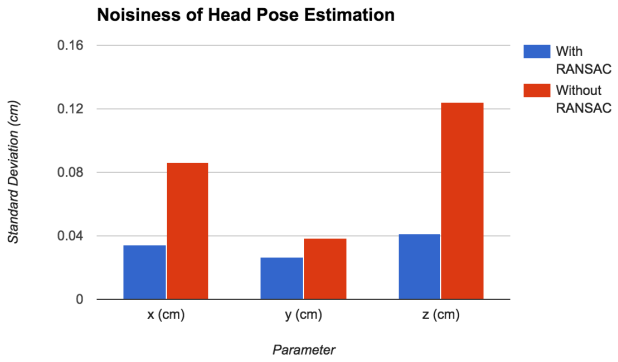

Figure 8. Standard deviations of the head position parameter estimates over 10 seconds, with RANSAC (blue) and without RANSAC (red).

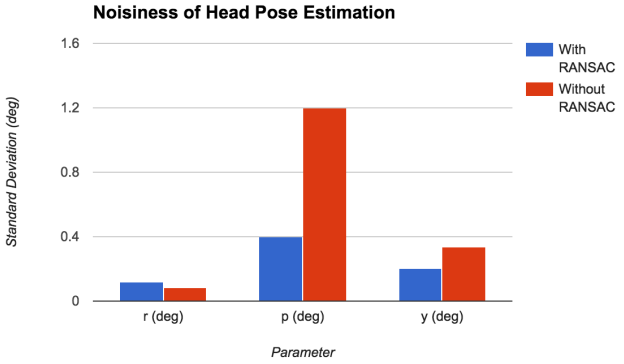

Figure 9. Standard deviations of the head orientation parameter estimates over 10 seconds, with RANSAC (blue) and without RANSAC (red).

When the user's head moved quickly, on-screen cursor position noticeably lagged behind as a result of the large values needed in the Kalman filter's measurement noise covariance matrix R_k required to stabilize cursor position; reducing the covariance parameter made cursor position unacceptably noisy. This result suggests that the tradeoff between responsiveness to new head pose measurements and robustness to noise in head pose measurements needs to be addressed by using a less noisy head pose estimation method or by collecting measurements at a higher sampling rate (e.g. 60 Hz rather than 30 Hz).

4.2. Quantitative Results

To characterize the noise robustness of our system, we measured the stability of various parts of our system while the user's head was held in a constant position which minimized the noisiness of dlib's facial landmark detector, pointed directly at the monitor for 10 seconds.

We found inherent limitations on the effectiveness of the dlib facial landmark detector. The locations of detected landmarks on images are often noisy. When a user holds their head still in front of the webcam while looking directly at the computer screen, the location of most facial landmarks on the user's head returned by dlib varies with a range of about 2 pixels, though some facial landmarks vary with a range of 3 to 4 pixels. This noise becomes increasingly severe for keypoints as the user's head pose deviates from this position. The landmark detector is particularly unstable at localizing the points forming the outline of the user's head when the user's head is significantly yawed. The detector always attempts to locate all 68 points in the image, even if some of the keypoints are occluded.

The errors from the dlib facial landmark detector were severely magnified in the triangulation step, with standard deviations of approximately 0.1 cm in the x and y directions and 0.4 cm in the z direction (Figure 6). Adding noise filtering to the estimated landmark positions significantly reduced the standard deviations in the z direction to be comparable to the standard deviations in the x and y directions, though some facial landmarks still experienced unacceptably high noise in the z direction (Figure 7).

Facial landmark position outliers in the z direction re-

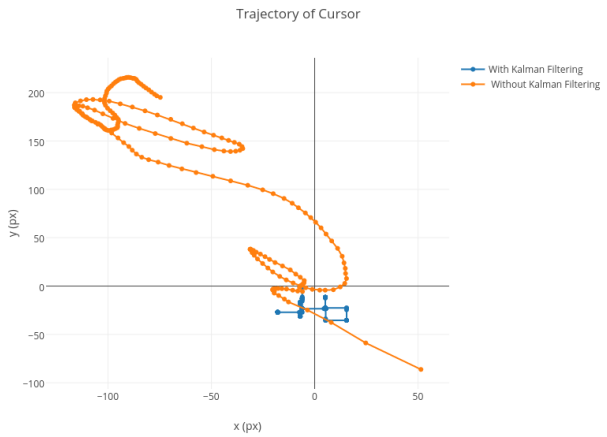

Figure 10. Trajectory of the cursor for a stationary head, without Kalman filtering (orange) and with Kalman filtering (blue).

mained in the triangulated point set even with noise filtering (Figure 7), and these outliers degraded the quality of the head pose estimation, with standard deviations of 0.12 cm in the z estimate, 0.08 cm in the x estimate, and 1.2 degrees in the head pitch estimate (Figures 8 and 9). The addition of RANSAC effectively removed the effect of these noisy outliers, reducing the respective standard deviations to levels comparable with those of other parameters. Nevertheless, the estimated head pose still suffered from noise which degraded the stability of the subsequent cursor localization step, and this noise was more severe when the user's head pose deviated from the optimal configuration used in these tests.

The cursor localization step was very sensitive to noise in R and T in the absence of Kalman filtering (Figure 10). With heavy Kalman filtering on the cursor position, the cursor position had standard deviations of 8 pixels in the x and y directions while the user's head was held still; this corresponded to a movement range of approximately 35 pixels in each direction. Thus, our system could correctly localize the cursor to the general region of the screen which the user's head was pointed at, but it did not exhibit the precision and stability for finer positioning (refer to the supplement for a video example).

5. Conclusion

In this paper, we implemented a real-time head pose estimation procedure with a stereovision webcam configuration. We used the estimated head pose to project a cursor on the screen which moves according to the direction of the user's head. Our results show that the cursor control task can be successfully completed, but there are still issues with stability and time delay.

There are a number of limitations with our approach which make it less than ideal for real world applications. Firstly, a stereo camera configuration requires multiple webcams and additional setup time. The webcams must be placed parallel to each other and close together enough to not occlude facial landmarks. Also, because all of the facial landmarks must be seen from every camera view, the user has a limited range of head direction. Another drawback is that the user must manually initialize the construction of a rigid head model prior to the pose estimation process. This can possibly be automated with a face detector that calibrates when a valid face is detected, but it still requires the user to hold their head stationary during calibration.

The primary area for future improvement is selection of a monocular head pose estimation algorithm with more favorable noise characteristics. Such an algorithm could be combined with our stereovision system for more accurate depth estimation than is attainable with a purely monocular algorithm and less noise than is exhibited with our stereovision system. Another direction for future work is to expand the system to track the head pose of multiple people. We can also optimize the current algorithm by testing under varied lighting conditions and camera resolutions.

6. Code and Supplements

Our code can be found at: https://github.com/ethanjli/CS231A-Screen-Stabilization

A video demonstration of our system responding to movements from the user's head can be seen at https://www.youtube.com/watch?v=ASCn2X2O9mI.

A video example of the noise exhibited by our system while the user's head is not moving can be seen at https://www.youtube.com/edit?o=U&video_id=u1tGTq_NJwU.

References

[1] G. Bradski. Opencv. *Dr. Dobb's Journal of Software Tools*, 2000. 1

[2] S. Gurbuz and E. Öztop and Naomi Inoue. Model free head pose estimation using stereovision. *Pattern Recognition*, 45:33–42, 2012. 2

[3] J. Hannuksela. Facial feature based head tracking and pose estimation. 2004. 1

[4] P. Henríquez, O. Higuera, and B. J. Matuszewski. Head pose tracking for immersive applications. *2014 IEEE International Conference on Image Processing (ICIP)*, pages 1957–1961, 2014.

[5] P. Jiménez, J. Nuevo, and L. M. Bergasa. Face pose estimation and tracking using automatic 3d model construction. *2008 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, pages 1–7, 2008. 1

[6] S. D. B. K Somani Arun, Thomas S Huang. Least-squares fitting of two 3-d point sets. *IEEE Transactions on pattern analysis and machine intelligence*, pages 698–700, 1987. 3

[7] D. E. King. Dlib-ml: A machine learning toolkit. *Journal of Machine Learning Research*, 10:1755–1758, 2009. 1, 2

[8] D. Lee. Stereovision: Library and utilities for 3d reconstruction from stereo cameras. 2016. 2

[9] S. Lemaignan, F. Garcia, A. Jacq, and P. Dillenbourg. From real-time attention assessment to "with-me-ness" in human-robot interaction. In *HRI*, 2016. 2

[10] S. Lemaignan, F. Garcia, A. Jacq, and P. Dillenbourg. From real-time attention assessment to with-me-ness in human-robot interaction. In *Proceedings of the 2016 ACM/IEEE Human-Robot Interaction Conference*, 2016. 1

[11] L.-P. Morency, A. Rahimi, N. Checka, and T. Darrell. Fast stereo-based head tracking for interactive environments. In *FGR*, 2002.

[12] E. Murphy-Chutorian, A. Doshi, and M. M. Trivedi. Head pose estimation for driver assistance systems: A robust algorithm and experimental evaluation. *2007 IEEE Intelligent Transportation Systems Conference*, pages 709–714, 2007. 2

[13] E. Murphy-Chutorian and M. M. Trivedi. Head pose estimation in computer vision: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31:607–626, 2009. 1, 2, 4

[14] A. Saeed, A. Al-Hamadi, and A. Ghoneim. Head pose estimation on top of haar-like face detection: A study using the kinect sensor. In *Sensors*, 2015.

[15] M. Ubilla, D. Mery, and R. F. Cádiz. Head tracking for 3d audio using the nintendo wii remote. In *ICMC*, 2010. 2

[16] R. Valenti, N. Sebe, and T. Gevers. Combining head pose and eye location information for gaze estimation. *IEEE Transactions on Image Processing*, 21:802–815, 2012. 2

[17] Wikipedia. Human head — wikipedia, the free encyclopedia, 2017. [Online; accessed 9-June-2017]. 3