Art Critic: Multisignal Vision and Speech Interaction System in a Gaming Context

Michael Reale, *Student Member, IEEE,* Peng Liu, *Student Member, IEEE,* Lijun Yin, *Senior Member, IEEE,* and Shaun Canavan, *Student Member, IEEE*

***Abstract*—True immersion of a player within a game can only occur when the world simulated looks and behaves as close to reality as possible. This implies that the game must correctly read and understand, among other things, the player's focus, attitude towards the objects/persons in focus, gestures, and speech. In this paper, we proposed a novel system that integrates eye gaze estimation, head pose estimation, facial expression recognition, speech recognition, and text-to-speech components for use in real-time games. Both the eye gaze and head pose components utilize underlying 3D models, and our novel head pose estimation algorithm uniquely combines scene flow with a generic head model. The facial expression recognition module uses the Local Binary Patterns with Three Orthogonal Planes (LBP-TOP) approach on the 2D shape index domain rather than the pixel domain, resulting in improved classification. Our system has also been extended to use a pan-tilt-zoom camera driven by the Kinect, allowing us to track a moving player. A test game, "Art Critic", is also presented, which not only demonstrates the utility of our system but also provides a template for player/NPC (non-player character) interaction in a gaming context. The player alters his/her view of the 3D world using head pose, looks at paintings/NPCs using eye gaze, and makes an evaluation based on the player's expression and speech. The NPC "artist" will respond with facial expression and synthetic speech based on its personality. Both qualitative and quantitative evaluations of the system are performed to illustrate the system's effectiveness.**

***Index Terms*—Gaze tracking, head pose estimation, expression recognition, speech recognition, text-to-speech, gaming interaction.**

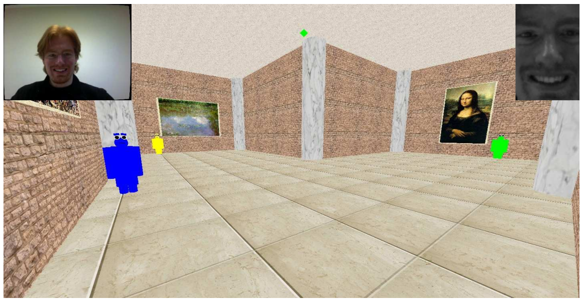


Fig. 1. The "Art Critic" game with multisignal vision and speech interface.

I. Introduction

THE principal goal of many games is to immerse the player in the world presented. When we encounter phenomena in the real world, however, our responses are often multifaceted, including our facial expression, body language, head pose, eye gaze, and speech. With the advent of modern game system inputs such as the Xbox Kinect [1] and the PlayStation Move [2], the industry has extended the gamer's inputs beyond the canonical controllers of the past and opened up the possibility of games responding to the visual and audio signals we send. In particular, a wealth of information can be found in the player's head pose and eye gaze. Once a region of interest is fixed upon, the player's facial expression potentially gives us the player's opinion of the object or person in focus. Of course, the player's speech should also be taken into account [3], [4], as it is perhaps the most direct form of human communication. With the exception of the Kinect system interface, however, automatic speech recognition and analysis are infrequently used in a gaming context. Many modern games, especially those within the role-playing and first-person shooter genres, attempt to simulate person-to-person interaction and communication as a critical part of the experience. One problem, however, is that the player generally feels somewhat disconnected from his/her character, since the player invariably chooses what they will say and how they will say it from a list of options. Worse, in some games the player then watches and listens to their own character make the response chosen. We believe that allowing the player to be more involved in the experience will increase immersion within the game and, consequently, make the game more enjoyable. It falls, then, to the next generation of games to read and interpret all channels of human communication and to respond appropriately. However, to the best of the authors' knowledge, there is no game or gaming system to date that incorporates head pose estimation, eye gaze estimation, facial expression recognition, speech recognition, and synthetic speech generation all together as part of the player's interactive experience. Therefore, we propose a multisignal vision and speech system to read, recognize, and respond to all these channels in a gaming context.

The automatic interpretation of data from any of these channels remains challenging. The overwhelming majority of eye gaze estimation approaches rely on "glints" – reflections of light off the cornea [5]. However, eye gaze may also be determined from pupil or iris contours [6], ellipse-fitting approaches [7], [8], the distance between the iris center and certain reference points (e.g., the eye corners) [9], [10], eye region segmentation and pixel-wise matching with 3D

Manuscript received XXXX. This work was supported in part by the Air Force Research Laboratory (FA8750-08-1-0096), National Science Foundation (IIS-0414029 and 1051103), and New York State Science and Technology Office (C050040).

M. Reale, P. Liu, L. Yin, and S. Canavan are with the Department of Computer Science, State University of New York at Binghamton, Binghamton, NY, 13902. (Contact author: lijun@cs.binghamton.edu).

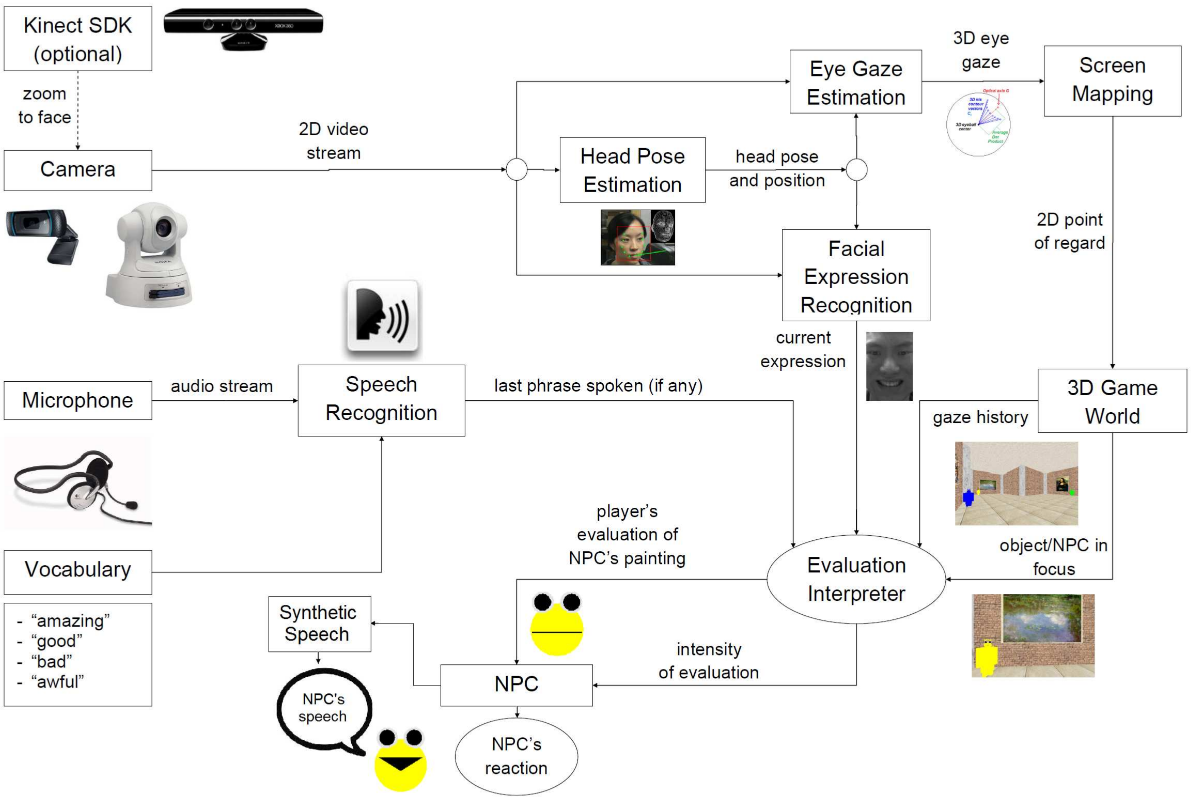


Fig. 2. Overview of multisignal vision and speech interaction system and application.

rendered eyeball models [11], [12], or even the reflection of the screen off the cornea [13]. Nonetheless, eye tracking in regular, non-infrared 2D imagery is still a difficult problem.

Some previous work shows successful head pose estimation based on a depth-aware camera [14] [15], multiple cameras [16], and a single camera [17], but accurate, continuous, and real-time head pose estimation remains a challenge, particularly with estimation of large head rotations.

There has been work done to incorporate facial expression recognition by itself into games, e.g., a multiplayer online game using Gabor wavelets and SVM for expression classification [18] and an HMM-based five expression recognition system for a network game [19]. Expression synthesis has been utilized in "serious games" [20] and storytelling systems [21], while both expression recognition and synthesis have been used for avatar synthesis [22], multimodal input systems [23], non-player character (NPC) behavior/personality customization [24], NPC emotional models [25], and virtual agents with gaze behavior [26]. However, the approaches and systems in this vein do not incorporate eye gaze, head pose, facial expression, speech information, and text-to-speech for interacting with the game world and its virtual inhabitants as we do.

Motivated by recent work [27], we propose a novel system with components performing eye gaze estimation, head pose estimation, facial expression recognition, speech recognition, and text-to-speech synthesis for use in real-time games. Based on our previous work [27], we utilize a 3D model of the eye for eye gaze estimation. A novel, subject-independent head pose estimation algorithm incorporating scene flow [28] and a generic 3D model is also presented. An active appearance model (AAM) based approach [29] is applied to detect and track 10 feature points on the player's 2D head images, and the head pose is then estimated using prior knowledge of the head shape and the geometric relationship between the 2D images and a 3D generic model. The expression recognition module leverages a unique 2D shape index [30] dynamic texture approach based on the Local Binary Patterns with Three Orthogonal Planes (LBP-TOP) algorithm [31]; user-specific templates are employed for each of the seven prototypic expressions (Angry, Disgust, Fear, Happy, Sad, Surprise, and Neutral). The camera images for the vision-based components can be from a simple webcam or from a pan-tilt-zoom camera driven by the Kinect SDK to focus in on the player's head. CMU's Pocketsphinx speech recognition library [32] is employed to recognize words and phrases spoken by the player, and the Festival text-to-speech library [33] allows NPCs to talk back to the player. To illustrate the power of such a system, a gaming application combining all of these components is presented. This game, "Art Critic" (Fig. 1), allows the player to navigate a 3D virtual art gallery and make evaluations of the paintings therein. Head pose changes are recognized and used to alter the player's view. Incorporating both head pose and eye gaze, the player's point of focus is tracked; in particular, the system checks whether the player has looked or is looking at a given painting in the gallery. An NPC "artist" stands next to its work. The player looks at the NPC and engages it in conversation. Then, when the player looks at either the painting or the artist, makes a facial expression, and speaks his/her evaluation of the painting (e.g., "good", "awful", etc.), the "artist" will react with both facial expression and speech based on the player's evaluation as well as on the NPC's personality. The strength of the verbal component of the evaluation (e.g., "good" vs. "amazing") will influence the intensity of the artist's response. Moreover, if the player has not even looked at the painting, the artist's reaction will be different. We perform a quantitative evaluation on the system in terms of facial expression and painting evaluation classification. We also conduct a qualitative evaluation through a questionnaire given to players asking about the experience of the game and how each component affected that experience.

Using multiple communication channels allows for more natural and comfortable interaction within the game. In contrast, a single-channel system would have to either assign certain functionality to more conventional input devices (i.e., mouse and keyboard) or make a potentially awkward gesture/control mapping scheme, such as having eye gaze control the player's view or having expression linked to movement commands. Moreover, our system makes intelligent use of all the important information from the face and head, thus forming a more complete interaction system rather than a conventional one with certain vision technologies attached as a bonus. Even though one could argue that certain information need only come from one channel (e.g., the overall evaluation could have come from the facial expression or the speech alone), using multiple channels allows for different combinations of signals, giving us complex, nuanced information from the player.

Our principal contributions are 1) a novel head pose estimation approach that couples scene flow with a generic 3D head model, 2) a unique, real-time system that makes use of head pose, eye gaze, facial expression, speech, and text-to-speech for intuitive, natural NPC and virtual world interaction in a gaming context, 3) exploration of the use of the LBP-TOP approach on the 2D shape index domain for improved facial expression recognition, and 4) an efficient implementation suitable for use in games. Fig. 2 shows an overview of our system and our application.

The paper is organized as follows: Section II presents our head pose estimation algorithm, Section III describes our eye gaze estimation approach, Section IV elaborates on our facial expression recognition component, Section V briefly discusses the speech recognition and text-to-speech components, Section VI introduces our game application "Art Critic" and presents our system evaluation results, and concluding remarks and discussion are given in Section VII.

II. Head Pose Estimation

We propose a novel head pose estimation approach that applies feature-based scene flow coupled with a 3D generic head model. This component uses a 2D video stream to determine head direction and position. First, the Viola-Jones approach [34] is applied to detect the frontal face. After the face is detected, the Active Appearance Model technique [29] is employed to detect and track predefined feature points on the face. The feature point coordinates in the 2D images are scaled and mapped to a 3D generic head model (described in Section II.B). Finally, based on the correspondence between the feature points, the 3D rotation angles can be calculated from the 2D coordinates by the so-called scene flow approach [28].

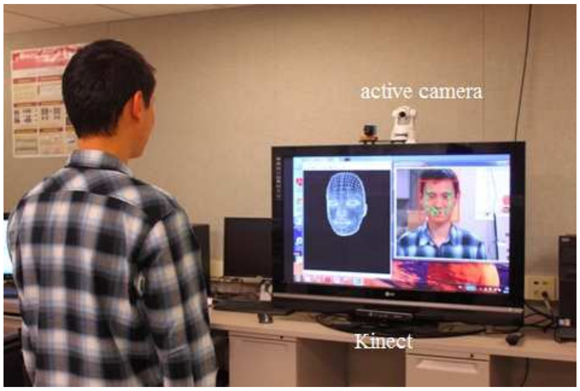
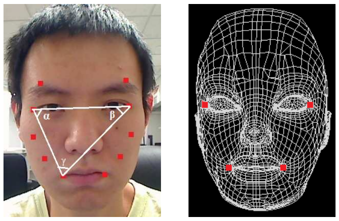


Fig. 3. Kinect and active camera setup.



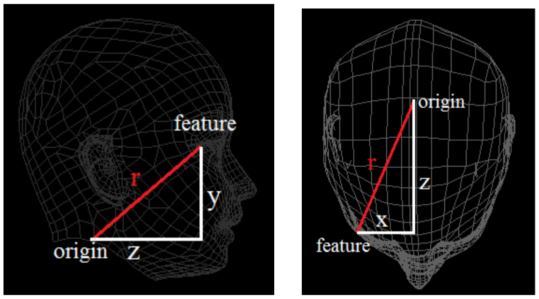
(a) (b)

Fig. 4. Feature points (a) on the 2D face and (b) on the 3D generic face model.

A. Face Region and Feature Detection

We have two possible camera setups. The first is a webcam sitting on top of a monitor, an arrangement ideal for computer gaming. Alternatively, the Kinect in conjunction with an active, pan-tilt-zoom camera may be used, which is more appropriate for console system games. Fig. 3 illustrates this dual-camera system.

The Kinect SDK provides functionality for body skeleton tracking; we use the 3D position of the head from this skeleton to control an active pan-tilt-zoom camera to rotate and zoom into the found head. The active camera (SONY SNC-RZ30N) can pan $\pm 170^\circ$ and tilt from -90° to $+25^\circ$ with a $25\times$ optical zoom lens. Thus, the system obtains a close-up view of the face. Please note, however, that the video from the active camera alone is used by our computer-vision-based system components, including our head pose estimation algorithm. The 3D position of the head from the Kinect is only used to drive the active camera, and otherwise it is not used directly in any of our system components. Consequently, the Kinect is not required for our system; as long as the player remains within the view of the camera (whether it is a webcam or a pan-tilt-zoom camera), our system can still be run. However, allowing



(a) (b)

Fig. 5. The side view and top view of the generic model with rotation radius.

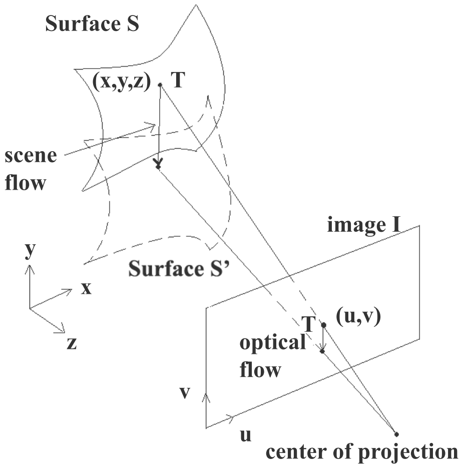


Fig. 6. The diagram of scene flow.

the Kinect to direct the camera permits the player to move within a typical console-gaming space (such as a living room). The reason we do not use imagery from the Kinect directly in our estimation algorithms is that the resolution is too low for a player moving around in a typical console-gaming space; that is, the face image would be too small. As such, we instead take advantage of the higher resolution a pan-tilt-zoom camera can provide, while leveraging the Kinect's ability to coarsely locate a player in the room.

Whether we use the video stream from a webcam or from an active camera, we refine the face area by applying an appearance-based technique based on the work by Viola and Jones [34] for face detection. Once the face area is identified, we use the classic active appearance model (AAM) approach [29] to detect and track the feature points on the face. A set of landmark images are used to create the training set. We defined ten 2D feature points on the face image, as shown in Fig. 4(a). These form a sort of horseshoe shape on the face and include the top points of the left and right eyebrows, the outer corners of the left and right eyes, four points on the cheeks, and the left and right corners of the mouth. These 2D landmarks are represented as a vector for training the shape and texture models by PCA.

B. 2D to 3D Coordinate Correspondences

In feature-based head pose estimation, the key step is to establish a set of geometric 2D-to-3D correspondences by matching 2D features to the 3D model features. The 3D generic model and the feature coordinates of the first frame of the 2D video sequence can be scaled and aligned by their scale factor m. This factor m is calculated by the distance between the feature points, such as the 2D distance Δu between the outer corners of the left and right eyes on the frontal face image and the corresponding 3D distance Δx on the generic model.

C. Head Pose Estimation Based on Feature Point Scene Flow

Similar to optical flow which is the two-dimensional motion of points in an image, scene flow is the three-dimensional motion of points in the 3D world space [28]. In theory, scene flow can be estimated by a complete knowledge of the surface geometry or by knowing image correspondences from multiple cameras [28]. It is impossible to estimate the 3D scene flow based only on one camera. In order to resolve this problem for our application, some restrictions and assumptions on the head rotation should be implied: in our case, we assume that head rotation and head translation do not occur simultaneously. Such an assumption is reasonable since, in practice, people seldom perform head rotation when their bodies move in translation a great deal.

We assume each feature point is moving on a surface which has a function $f(x, y, z; t) = 0$. Each feature point is rotated around the axis in the middle of the neck. Fig. 5 shows the origin and one feature point on the side view and top view of our generic head model. As shown in Fig. 6, the image point (u, v) is the projection of the 3D point (x, y, z) by a projection matrix P. Thus, the feature point $(u, v; t)$ on a video sequence is the result of the projection of the corresponding feature point on the 3D surface f at the 3D point $(x, y, z; t)$ of the 3D motion object. Therefore, a scene flow of a point T in the 3D space generates an optical flow of the corresponding point in the 2D image domain.

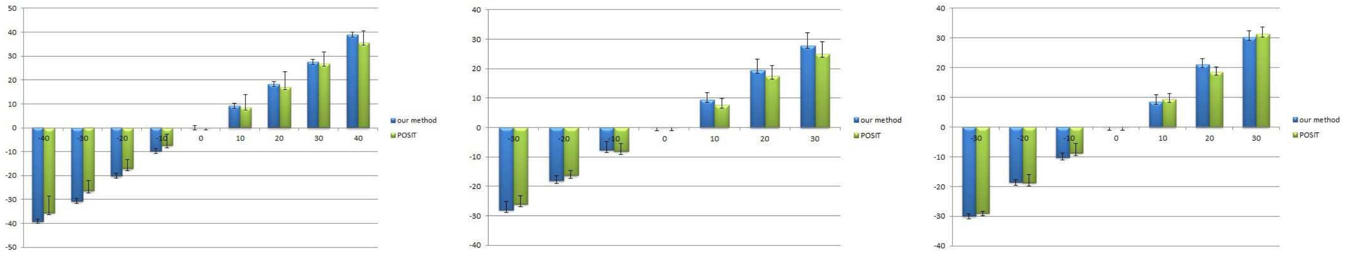
Let $\boldsymbol{x}(t) = (x, y, z; t)$ be the 3D path of a feature point on the face surface, and let $\boldsymbol{u}(t) = (u, v; t)$ be the corresponding feature point in the image. As the feature point $\boldsymbol{x}(t)$ moves with head rotation, we assume that its rotation radius $r = r(\boldsymbol{x}(t), t)$ remains constant; that is

$$\frac{dr}{dt} = 0 \tag{1}$$

The rotation radius of feature points is calculated by

$$r = \sqrt{x^2 + y^2 + z^2} \tag{2}$$

Equation 3 describes how the 3D coordinates of the feature points are estimated:



(a) Yaw

(b) Pitch

(c) Roll

Fig. 7. Mean and standard deviation of estimates for yaw, pitch, and roll rotations for our approach and POSIT with the webcam tests (units in degrees).

$$\begin{cases} x_n = x_o + \sum_{i=1}^{n} \Delta x_i \\ y_n = y_o + \sum_{i=1}^{n} \Delta y_i \\ z_n = z_o + \sum_{i=1}^{n} \Delta z_i \end{cases} \tag{3}$$

where (x_n, y_n, z_n) is the estimated coordinate of the feature point of frame n, (x_o, y_o, z_o) is the original coordinate of the generic model, i is the index of the frame number, and n is the total number of frames since initialization. $\Delta x_i, \Delta y_i, \Delta z_i$ are calculated by:

$$\begin{cases} \Delta x_i = m \times \Delta u_i \\ \Delta y_i = m \times \Delta v_i \\ \Delta z_i = \frac{-x_o}{\sqrt{r^2 - x_o^2 - y_o^2}} \Delta x_i + \frac{-y_o}{\sqrt{r^2 - x_o^2 - y_o^2}} \Delta y_i \end{cases} \tag{4}$$

We define four feature points on the 3D generic face model to calculate the normal vector. These are the outer corners of left and right eyes, and the left and right corners of mouth, as illustrated in Fig. 4(b). Note that these four points on the 3D generic model are a subset of the ten 2D points described earlier. A triangle is formed by the two outer corners of the left and right eyes and the middle point of the two mouth corners. The normal vector of such a triangle is relatively expression invariant, thus representing the pose orientation of a head accordingly. Although the mouth corners are sensitive to facial expression, we use the average value of the two mouth corners. Normally, when people perform facial expressions, the mouth corners move symmetrically. Even if the AAM fails to detect the exact location of the mouth corners due to expression changes, using the average of the two mouth points means the head pose estimation error caused by facial expression would be fairly small.

To address the issue of scene flow drift, we reinitialize the (x_o, y_o, z_o) and radius r for each point as well as the scale factor m if the subject looks straight towards the camera. This is determined by comparing the angles of a triangle formed from the outer eye corner points and one of the mouth corner points with the corresponding triangle from the initialization frame. Fig. 4(a) shows an example triangle. α, β, and γ are the angles to be used for the drift reduction. Based on the property of similar triangles, if the corresponding angles between the first frame and the current subsequent frame are congruent, the two triangles must be similar, and we reinitialize the model data.

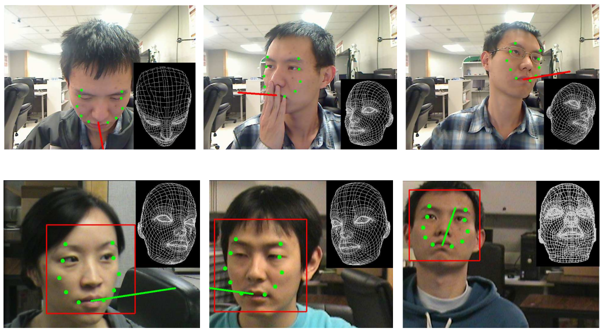


Fig. 8. Real-time head pose tests; (first row) tests on a Logitech webcam with pose vector given by red line; (second row) tests on the active camera with pose vector given by green line. The estimated head poses are verified by the follow-up rotation of a generic model driven by the pose parameters obtained from the live subjects.

D. Head Pose Evaluation

1) Test on video sequences captured from 2D cameras: As mentioned earlier, our system can be set up with a regular webcam or with an active pan-tilt-zoom camera. The first row of Fig. 8 shows some examples from a Logitech webcam, while the second row contains results from the pan-tilt-zoom camera. The system works with a resolution of 640×480 in real time. The green dots are the tracked feature points on the face; the estimated head pose vector is shown by a red line in the top row of Fig. 8 and a green line in the bottom row of Fig. 8. We have also transferred the estimated pose to the 3D generic model to visualize the head orientation. The generic model rotates in real time along with the subject's head. Subjectively, our system works fine in different imaging conditions, including when the face is partially occluded and when the subject is wearing eye glasses.

In order to objectively evaluate the accuracy of our head pose estimation algorithm, it is necessary to obtain the ground truth for the head poses in the test videos. To do so, we affix a laser pointer to each subject's forehead. This allows the subject to locate precisely what they are looking at in real time. In our experiments, we tested the performance on five subjects from our lab. The range of yaw estimation is $[-40°,\ 40°]$, in which the right side is positive. The range of pitch estimation is $[-30°,\ 30°]$, in which up is positive. The range of roll estimation is $[-30°,\ 30°]$, in which the right side is positive.

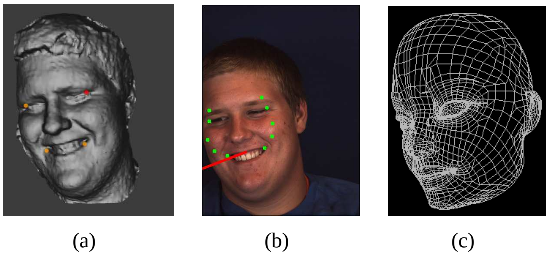


Fig. 9. Test on 3D dynamic video sequences with spontaneous head movements and expressions.

TABLE I
AVERAGE ERROR OF HEAD POSE ESTIMATES ON 3D VIDEO SEQUENCES

	Average Error (in degrees)		
Method	**Pitch**	**Yaw**	**Roll**
POSIT [35]	6.7	7.9	2.1
Our Method	3.8	6.2	1.4

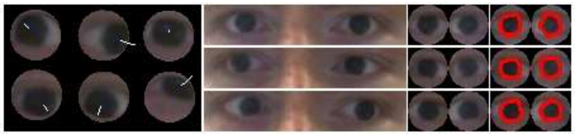


Fig. 10. (First column) 3D rendered eyeballs with white lines indicating optical gaze direction; (second column) original 2D image used for 3D iris detection; (third column) 3D eyeballs rendered with the iris looking into the camera; (fourth column) iris contours, shown as red dots between the iris and the rest of the eyeball, found on rendered eyeball image.

The charts in Fig. 7 show the average estimate and standard deviation of yaw, pitch, and roll, respectively. The average of the absolute value error for yaw is 4.83°, for pitch is 3.36°, and for roll is 1.34°. We also present the results from using the Pose from Orthography and Scaling with Iterations (POSIT) head pose estimation approach [35]. As one can see from Fig. 7, our approach performs better on average. The estimated results demonstrate the effectiveness of our pose estimation approach with 2D cameras. (Please see the supplemental material for a video demo.)

2) Test on video sequences captured from 3D cameras: We have also tested our algorithm using video textures from our 3D dynamic model database [36]. The 3D model sequences were captured by the *Di3D* system [37]. We tested on 3D video sequences of 40 subjects with spontaneous head movements and various expressions.

In order to evaluate the accuracy of our head pose estimation algorithm, we generated comparison model data from the 3D model sequences directly. Based on the 3D tracking software provided by the *Di3D* imaging system [37], we are able to track feature points across the 3D model sequences. This feature point tracker relies on the 3D mesh data, and ergo it is more accurate and stable than a 2D-based approach. Fig. 9(a) shows the four feature points defined and tracked on the 3D model surfaces. Based on those 3D feature points, we used the same approach described earlier (Fig. 4(b)) to generate the head orientation vector, giving us a reference pose orientation for each frame of each video to use for comparison.

Given the comparison head pose data, we estimate the difference between the estimated head poses from video textures and the comparison poses from the corresponding 3D models. Fig. 9 shows one example of a 3D model sequence with tracked feature points. Fig. 9(a) is the captured 3D model, Fig. 9(b) is the corresponding texture, and Fig. 9(c) is the generic model rotated by the estimated head orientation from the textures. Table I shows the average errors of pose estimation from 4,279 frames of the 3D video database in terms of pitch, yaw, and roll, respectively. We again compare our work to results from POSIT [35], and our approach does noticeably better.

The experimental results show that errors occur more often with yaw rotation than with other rotations. The spontaneous facial behavior data does include some cases of the users rotating their heads while performing translation in the x axis. As a result, this translation with rotation brings some error into our yaw estimation results. Overall, however, our results are still promising. POSIT relies on all the feature points of face to make its pose estimate; as such, any non-neutral expression on the face can greatly influence the estimation results. Moreover, when the face rotates, some feature points on the edge of the model can be less reliable and thus affect the POSIT results. Our approach relies on fewer points that are more robust to expression changes. Thus, this experiment demonstrates that our algorithm is in general applicable to spontaneous head movements with various facial expressions.

III. EYE GAZE ESTIMATION

A. Iris Detection and Contour Extraction

Based on existing work [27], we determine the current eyeball positions by offsets from the 3D head pose and position. These offsets are calculated from a calibration procedure which is described in [27]. The eye detection algorithm maps the current camera image as a 2D texture onto the current positions of the 3D eyeballs, rotates the eyeballs in pitch and yaw, renders the rotated eyeballs, and picks the rotated eyeballs that look most like the user is looking into the camera. This is evaluated by 1) computing the absolute pixel intensity difference of the center region of each rendered eyeball from a dark, circular template and 2) circle-fitting on the gradient magnitude image. We use CUDA [38] to determine the scores for multiple eyeball images simultaneously. Once the best eyeball rotations and scales are determined, the eyeballs are rotated back and projected into image space, giving us our 2D iris centers. The first column of Fig. 10 shows some sample 3D eyeballs rendered at different angles. For iris contour extraction, we effectively shoot rays outwards from the center of each optimally-rendered eyeball image, similar to [39]. Initially, the points along the rays with the highest dark-to-light gradient value going outwards within a certain radius range are chosen, and then all points falling outside of a more restrictive range are eliminated to remove eyelid and specular highlight points. The right-most column of Fig. 10 shows some examples of iris contours. A GLSL pixel shader [40] is leveraged here to make the contour extraction more efficient.

To eliminate eyelid points, the mapped iris contour points are rotated in line with the head pose direction and iteratively

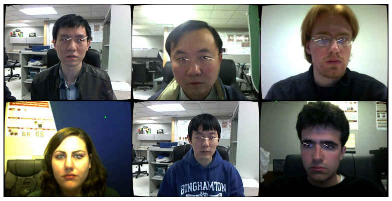


Fig. 11. Sample gaze results from different subjects.

grouped until a collection that fits a plane parallel to the head pose x axis and going through the eyeball center is found. Two such “eyelid” planes are found, one for the upper and one for the lower eyelid. Contour points outside of or to close to the eyelids are eliminated, and the entire eyelid-finding procedure is performed twice for each eyelid on each eye.

B. Gaze Estimation

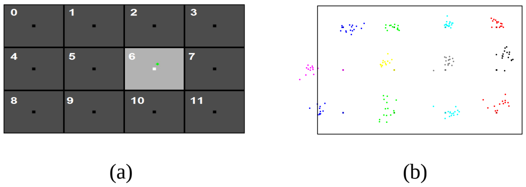
Each 2D contour point is converted to a 3D world vector, intersected with the current eyeball sphere, converted to a vector from the eyeball center, and normalized to give us an “iris contour vector” C_i. It is assumed we also have the iris radius, stored as an expected dot product d between the optical axis G and each contour vector C_i. Therefore, to estimate the optical axis G, one solves a system of linear equations as defined in [27]. If we take V to be the normalized vector from the eyeball center to the iris center point mapped onto the eyeball surface, the basic idea is to find each eye’s optical axis G such that 1) it is parallel to V and 2) the dot product of G and each C_i is d. Note that d, V, and the constant 1 are repeated in their respective matrices N times, once for each contour vector. Doing so gives equal weight to our two constraints. Once G is found, it is normalized. To get the visual axis, a fovea offset computed during the calibration procedure is used. The fovea offset is rotated based on the rotation angles of the optical axis G. The optical axis is then intersected with the eyeball sphere to get a new estimate for the 3D iris center, and the normalized vector from the fovea to this new iris center is the final gaze direction for the given eye.

The procedure above is performed for each eye independently. Then, the averages of the two foveae and the visual axes are used as the final starting point and gaze direction, respectively. Assuming the screen’s 3D position, size, and orientation are already known, a simple ray-plane intersection gives us the 2D gaze point of regard.

Fig. 11 shows some sample gaze estimation results. (Please see the supplemental material for a video demo.)

C. Eye Gaze Evaluation

We performed a real-time gaze and point-of-regard estimation experiment with a webcam wherein each user was asked to look at 12 gaze markers on the screen (effectively, the center of each brick in a 3×4 grid, as shown in Fig. 12(a)). The user focused on each marker for 2-4 seconds. We recorded the angular error, which is measured as the angle between the estimated gaze direction vector and the vector from the gaze starting point to the target point. We also recorded the “hit percentage,” which refers to how frequently the cursor was within the target block. Please note that a point going past the edge of the screen was still considered a “hit” on the gaze target block closest to the gaze point. Given our application, this is reasonable since it is assumed the user is looking somewhere on the screen while using the system. With 4 subjects, the overall angular error was 5.953°, and the average hit percentage was 90.54%. The error is relatively low for a natural light eye gaze estimation approach [5].



(a) (b)

Fig. 12. (a) Gaze test grid. Markers glow white when active, box becomes light gray when eye cursor enters region. Eye gaze cursor is the green diamond. The numbers are drawn here for clarity but were not drawn during the test. (b) Example of redrawn gaze points from one of our tests (the black border around the gaze points is the boundary of the screen region).

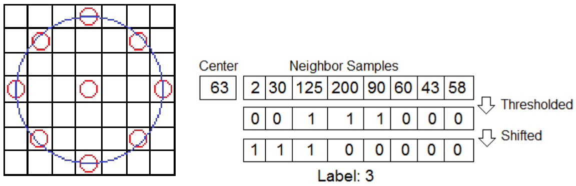


Fig. 13. Sample uniform LBP feature with 8 samples and radius of 3.

IV. Facial Expression Recognition

A. 2D Shape Index Based Dynamic Textures

Dynamic textures (DT) encode texture information across space and time. In this case, these textures are constructed with concatenated Local Binary Pattern histograms from Three Orthogonal Planes (LBP-TOP) [31]. Basically, for a given image sequence, the LBP histogram for the middle image in the time sequence is computed to give us the XY plane histogram. With the X coordinate set to its center value, an “image” plane is constructed with all variations of Y and T (time) to give us the YT plane, and the LBP histogram is extracted from that as well. A similar process is performed for the XT plane. The histograms for each plane are normalized individually, and the concatenated histograms describe the texture in three dimensions. To reduce histogram size, we use only the uniform LBP features [31], with sample counts of 8 and radii of 3 for all dimensions. Fig. 13 shows a sample uniform LBP feature with 8 samples and radii of 3 in x and y.

For facial expression recognition, the head position is first determined. Then, the face image is scaled based on the 2D eyeball centers. The image is then broken up into 9 by 8 overlapping blocks with an overlap ratio of 70%; each block has its own dynamic texture (DT) histogram. All DT block histograms are concatenated together to form one feature vector describing the entire face region. Every frame, a time

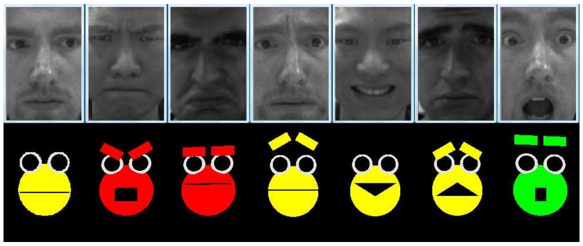


Fig. 14. Subject (top row) and NPC (bottom row) expressions from left to right: Neutral, Angry, Disgust, Fear, Happy, Sad, Surprise.

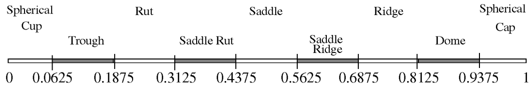


Fig. 15. Nine well-known shape types and their locations on the S_I scale [30].

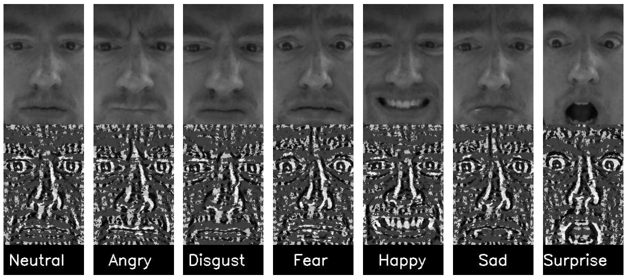


Fig. 16. Grayscale and shape index images for each expression.

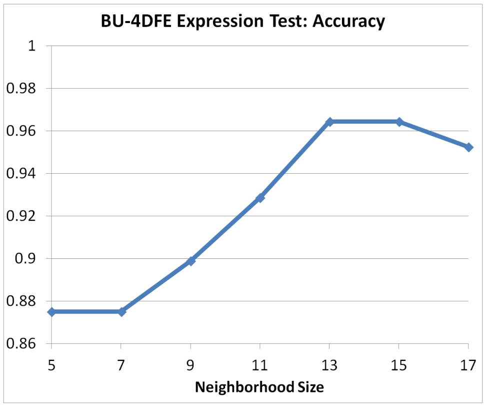


Fig. 17. Shape index neighborhood size vs. accuracy on BU-4DFE test.

slice of the last 30 frames is used for classification. Note, however, that we differ from [31] in three ways in order to improve the recognition performance. Firstly, the system extracts the DT histogram for each user's expression and saves it as a template. For each new frame's DT histogram, it is compared to each template using the log-likelihood statistic:

$$L(T,M) = -\sum_{b=1}^{B} T_b \log M_b \tag{5}$$

where B is the number of bins, and T_b and M_b correspond to the sample and model probabilities at bin b, respectively. If M_b equals zero, we add nothing to the entropy to avoid unfairly penalizing vectors that contain some zero components in their model probability vectors. When the system starts and the subject's face is found, the LBP histogram for a single frame is also extracted, and the nearest match is found in a database of known subjects. If the subject confirms that they are the match found, the user's expression histograms and eye gaze calibration data are loaded. Otherwise, the system prompts for the user's name. The user must then perform each of the seven prototypic expressions (Angry, Disgust, Fear, Happy, Sad, Surprise, and Neutral), and the system will record the templates for that specific user. When the application is closed, the new subject-specific data will be saved and can be reloaded in the future. Secondly, due to the good characterization of facial expressions using topographic features [41], the 2D shape index images are computed as input into the LBP-TOP algorithm. Another motivation is that shape index images will be relatively robust to different lighting conditions. One shape index image is generated per frame. Thirdly, we have found that, in practice, having the user continue to perform the expression through the entire recorded sequence achieves more stable performance. The top row of Fig. 14 shows each of the expressions performed.

The shape index image is computed as follows: if the grayscale image is treated like a 3D height map, the local neighborhood of each pixel can be fitted to a cubic polynomial as described in [42]. The principal curvature directions and magnitudes can be found from the Weingarten matrix formed from the cubic polynomial [42]. The shape index of a given point describes the nature of the area around that point [30]. Equation 6 demonstrates how to compute the shape index around a point p:

$$S_I(p) = \frac{1}{2} - \frac{1}{\pi} \arctan \frac{\kappa_1(p) + \kappa_2(p)}{\kappa_1(p) - \kappa_2(p)} \tag{6}$$

where κ_1 and κ_2 are the principal curvatures of the surface, with $\kappa_1 \geq \kappa_2$ [30]. Note that the $\arctan$ function in this equation returns an angle in radians. The shape index values are in the range $[0,\ 1]$; these values can in turn can be scaled to the range $[0,\ 255]$ and thus treated as another image. This image can be used instead of the original face image in our system. Fig. 15 shows the shape index scale with some well-known shape types, while Fig. 16 shows some example shape index images for each expression.

To ensure performance acceptable for gaming, we adopt three strategies. First, both the shape index computation and LBP extraction stages are performed for each pixel in CUDA [38]. Second, the LBP features for the entire XY plane are computed, and the correct LBP data is copied to each of the blocks. Third, for the XT and YT planes, only the LBP features for the new incoming data (that is, the part of each plane with the most recent t coordinate) are computed from each frame, and the histogram is updated accordingly.

B. Facial Expression Recognition Evaluation

We evaluate the expression recognition performance on the BU-4DFE database [43]. We chose 24 subjects and marked the onset, peak, offset, and ending frames of each expression sequence. On average, each expression video is about 100

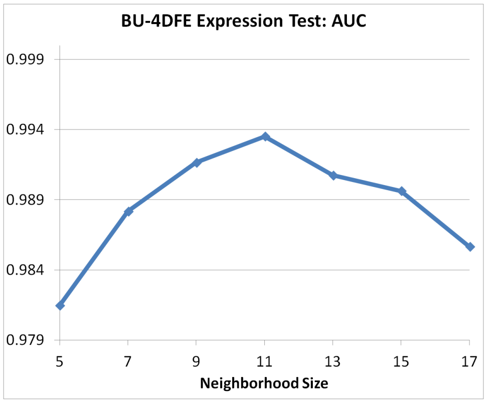


Fig. 18. Shape index neighborhood size vs. AUC on BU-4DFE test.

TABLE II
BU-4DFE Facial Expression Classification Results using Shape Index Images with Neighborhood Size 13 × 13 (Acc. = Accuracy, Pre. = Precision, Rec. = Recall, F1 = F1 Score, AUC = Area Under Receiver Operating Characteristic curve, and W. Avg. = Weighted Average)

Class	Acc.	Pre.	Rec.	F1	AUC
Angry	1	1	1	1	0.999
Disgust	0.958	1	0.958	0.979	0.995
Fear	1	0.923	1	0.96	0.997
Happy	1	0.96	1	0.980	1
Sad	1	0.889	1	0.941	0.999
Surprise	1	1	1	1	1
Neutral	0.792	1	0.792	0.884	0.946
W. Avg.	**0.964**	**0.967**	**0.964**	**0.963**	**0.991**

TABLE III
BU-4DFE Facial Expression Confusion Table using Shape Index Images with Neighborhood Size 13 × 13

Classified as →	Angry	Disgust	Fear	Happy	Sad	Surprise	Neutral
Angry	24	0	0	0	0	0	0
Disgust	0	23	0	1	0	0	0
Fear	0	0	24	0	0	0	0
Happy	0	0	0	24	0	0	0
Sad	0	0	0	0	24	0	0
Surprise	0	0	0	0	0	24	0
Neutral	0	0	2	0	3	0	19

TABLE IV
BU-4DFE Facial Expression Classification Results using Regular Grayscale Images

Class	Acc.	Pre.	Rec.	F1	AUC
Angry	1	0.923	1	0.96	0.999
Disgust	1	0.96	1	0.980	1
Fear	1	0.828	1	0.906	0.994
Happy	1	1	1	1	1
Sad	0.958	0.885	0.958	0.92	0.992
Surprise	0.958	1	0.958	0.979	0.992
Neutral	0.625	1	0.625	0.769	0.964
W. Avg.	**0.935**	**0.942**	**0.935**	**0.930**	**0.992**

frames long. Each subject was chosen based on two criteria. First, each of the subject's expression sequences had to contain a minimum of 45 frames between the peak and ending frames. The first 30 frames are used as the training sequence for the peak expression, and the last 30 frames are used for the testing sequence. This ensures that the training and testing sequences would only overlap at most by half. Second, each subject had to have at least two sequences with a minimum of 5 Neutral frames each. The expression videos do not generally have Neutral segments that are 30 frames in length. Therefore, the Neutral training and testing sequences were created by going backwards and forwards through each 5-frame Neutral sequence until 30 frames were filled. Thus, we had a total of 5,040 frames for training and the same number for testing. The size of each face image was scaled to 160×240 pixels.

We compute the accuracy, precision, recall, and F1 score for each class. Weighted averages are used for the overall statistics. Accuracy refers to the number of correctly-classified members of a given class (true-positives) over the total number of members in the class. Precision is defined as the number of true-positives for a class over the number of samples classified as that class (in other words, both true-positives and false-positives). Recall for a class is the number of true-positives over the number of true-positives and false-negatives (the latter being samples incorrectly classified as not belonging to the given class). F1 score is the weighted average of precision and recall, computed as $2 \cdot \frac{precision \cdot recall}{precision + recall}$.

In addition to these statistics, we also compute the Area Under Receiver Operating Characteristic curve (AUC) scores per class and use a weighted average of the scores to get the final AUC score. The rationale for computing these scores is that they give us theoretical upper-bounds on the classification performance. To ensure that the log-likelihood scores per class were comparable across different samples, the scores are normalized for each sample.

Tests were conducted using LBP-TOP on shape index images generated using different neighborhood sizes (i.e., varying N when the neighborhood around each pixel was $N \times N$ in size). Fig. 17 illustrates the expression recognition accuracy as the neighborhood size varies, while Fig. 18 shows the expression recognition AUC as the neighborhood size varies.

As the chart shows, our best results in terms of accuracy were with a neighborhood size of 13×13, giving us an accuracy of 96.4% and an AUC score of 0.991. Table II shows the classification results using the shape index images with a 13×13 neighborhood. Table III is the confusion matrix for the best shape index results.

The results for each class are fairly high, and the accuracy for Neutral is acceptable. One might note the drop in accuracy after $N = 15$ and the drop in AUC score after $N = 11$. The reason for this is that, if the shape index neighborhood becomes too large compared to the image size, important information begins to get smoothed over because of the polynomial fitting. That is, after a certain point, the polynomial approximation of the face surface becomes less accurate and begins to filter out relevant data as well as noise.

For comparison, we also performed a test using LBP-TOP on regular grayscale images; the results are given in Table IV. Our accuracy results using shape index imagery are about 3% higher, mostly due to a decided drop in the classification accuracy of Neutral using regular grayscale imagery. Also, our AUC results are comparable. Indeed, the highest AUC score

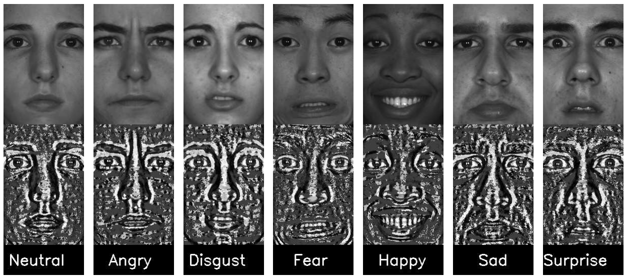


Fig. 19. Examples of grayscale and shape index images for correctly-classified samples from BU-4DFE using shape index imagery.

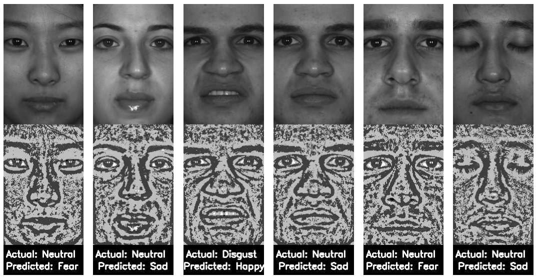

Actual: Neutral Predicted: Fear
Actual: Neutral Predicted: Sad
Actual: Disgust Predicted: Happy
Actual: Neutral Predicted: Sad
Actual: Neutral Predicted: Fear
Actual: Neutral Predicted: Sad

Fig. 20. Grayscale and shape index images for the misclassified samples from BU-4DFE using shape index imagery.

possible using shape index imagery was 0.994 with neighborhood size 11×11, although the corresponding accuracy is lower under those conditions.

Overall, these results demonstrate that this approach is appropriate for our system. Please see Fig. 19 for examples of correctly classified samples from BU-4DFE using shape index images. Fig. 20 shows the misclassified samples using shape index imagery. Almost all of the misclassified cases are Neutral; however, it can be observed that some of them do not look entirely expressionless, particularly the samples misclassified as Sad.

V. Speech Recognition and Text-to-Speech

To recognize the user's verbal evaluations and to allow the system to respond with speech, our system has both speech recognition and text-to-speech components. The speech recognition module makes use of CMU's Pocketsphinx software [32], and the text-to-speech module uses the Festival library [33]. The speech component starts listening as soon as the user begins speaking and stops when the user is silent for more than 1 second. It then extracts Mel-frequency cepstral coefficients (MFCCs) [44] to form the feature vector for the given audio sequence. Given the feature vector, an acoustic model is used to find the "senones" (effectively a complex phone or class of sounds), while a dictionary maps these senones to words. A language model can help filter out highly improbable word sequences [32].

For our application, our dictionary includes 5 words: "hello", "awful", "bad", "good", and "amazing". Please note, however, that our dictionary can be very easily expanded. Also, the existing module already returns a transcription of the complete spoken phrase; thus, future work could involve extending our recognition system to interpret full sentences.

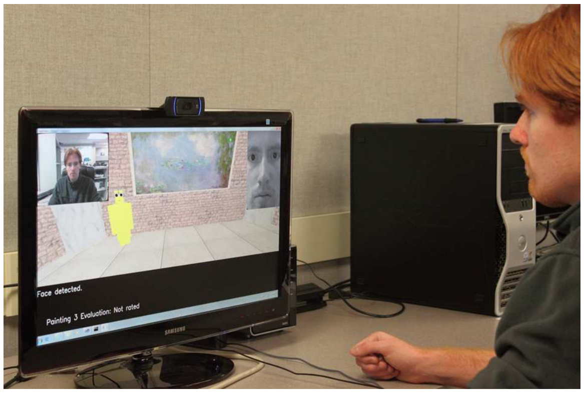


Fig. 21. "Art Critic" game application in action.

The speech recognition module runs concurrently with the main program as a separate thread, allowing simultaneous recognition of all signals. It is only paused when the NPC is speaking (to prevent a sort of feedback loop wherein NPC speech is mistaken for player speech).

VI. Validation through a Gaming Application: "Art Critic"

A. Application Overview

To demonstrate the utility of our complete real-time system in a gaming context, we have developed a game wherein the player is an art critic; Fig. 21 shows the application in action. The player can walk around an art gallery with paintings on the walls and NPC "artists" standing next to each painting. The player's head pose is used to alter his/her view: when the head's yaw or pitch is past a certain threshold, the player's view rotates accordingly. Eye gaze is also tracked, and the system notes not only whether the player is currently looking at a given NPC or painting but also how long the player has looked at each painting. If the player's distance from a painting exceeds a certain threshold, however, the system will intentionally not record their gaze behavior, since the player is too far away to really see the painting properly. The frame-to-frame facial expression is stored in a history of N frames (where N can be up to 60), and the current facial expression is the expression with the highest number of instances in the frame history (i.e., a majority voting scheme). The eye gaze target is handled in the same fashion. This increases the robustness of the system overall.

The player then looks at an NPC and says "Hello". The NPC responds and inquires whether the player has seen its work (if the player has already looked at the painting for some time, the NPC will instead note that the player has been looking at its painting). The player then looks at either the NPC or its painting, makes a facial expression, and gives a one-word evaluation ("good", "awful", etc.). Different combinations of facial expressions and speech will be interpreted as different overall evaluations, as shown in Table V(a). For example, an Angry expression with the word "bad" will simply indicate Dislike; however, a Happy expression with the word "bad" will be viewed as Mocking. Moreover, different words indicate different intensities (e.g., "awful" is stronger than "bad"). The

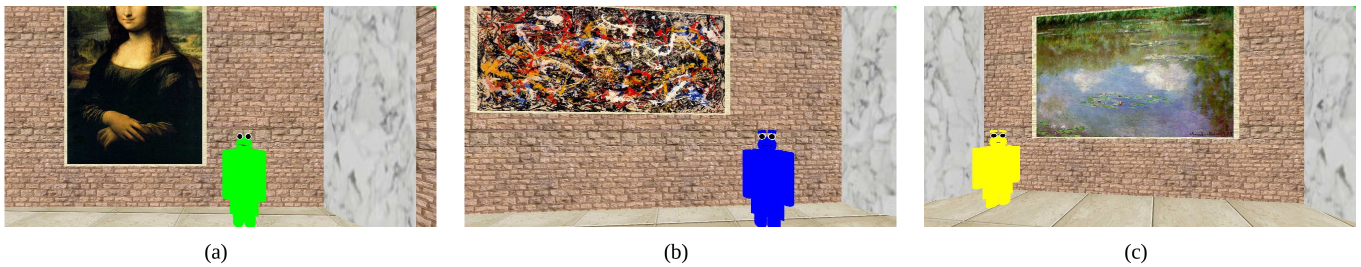


Fig. 22. "Art Critic" game NPC artists. (a) The Cheerful artist; (b) the Miserable artist; (c) the Cowardly artist.

TABLE V
(A) PLAYER EVALUATIONS, AND (B) NPC'S EMOTIONAL RESPONSE BASED ON NPC'S PERSONALITY AND PLAYER'S EVALUATION (RELATIVE INTENSITIES SHOWN; ABSENCE OF INTENSITY LABEL INDICATES STRONG INTENSITY; "EVAL." REFERS TO PLAYER'S EVALUATION, WHILE "NPC PER." REFERS TO THE NPC'S PERSONALITY; "WEAK " $<$ "MILD" IN TERMS OF INTENSITY.)

(a) Player Evaluations

Speech / *Expression*	**"bad/awful"**	**"good/amazing"**
Angry/Disgust	Dislike	Envy
Fear	Polite Dislike	Like
Happy	Mocking	Like
Sad	Polite Dislike	Like
Surprise	Shock	Awe
Neutral	Dislike	Like

(b) NPC's Emotional Response

NPC Per. / *Eval.*	**Cheerful**	**Miserable**	**Cowardly**
Dislike	Neutral	Sad	Sad
Mocking	(Mild) Angry	(Mild) Angry	(Mild) Fear
Polite Dislike	(Weak) Happy	(Mild) Sad	(Mild) Sad
Shock	(Weak) Fear	Surprise	Fear
Like	Happy	(Mild) Happy	(Mild) Happy
Envy	(Weak) Sad	(Weak) Disgust	(Mild) Fear
Awe	Happy	Surprise	(Mild) Surprise
Dismissive	(Weak) Angry	Angry	Sad
No evaluation	(Weak) Happy	(Weak) Sad	(Weak) Fear

artist will react with its own facial expression as well as with audible speech, and the artist's reaction will be scaled by how intense the player's evaluation was. Please note this intensity is only from the word used by the player, not from the player's facial expression intensity. Finally, whether the player gazed at the painting long enough to make a fair evaluation is considered; for example, if the player barely looked at a painting and gives it a negative evaluation, the artist will interpret that as being (unfairly) Dismissive.

The NPCs react to the player's evaluation based on their personalities. Fig. 22(a) shows the "Cheerful" artist, Fig. 22(b) shows the "Miserable" artist, and Fig. 22(c) shows the "Cowardly" artist[1]. The color of the NPC is used to indicate its personality (i.e., blue for "Miserable", yellow for "Cowardly", and green for "Cheerful"). The bottom row of Fig. 14 shows each of the possible NPC responses with maximum intensity, while Fig. 23 illustrates each NPC facial expression with the possible intensities. Again, however, the intensity of the artist's response will be influenced by the intensity of the player's evaluation. The complete list of evaluations and responses is shown in Table V(b). The relative NPC response intensities shown indicate the maximum possible response intensity. One can see, for example, that the Cowardly artist is somewhat afraid of an envious player, perhaps fearing for its own safety or the safety of its painting. The Cheerful artist, in contrast, exhibits slight sadness, indicating that it pities the player for being jealous of its work. (Please see the supplemental material for a video demo.)

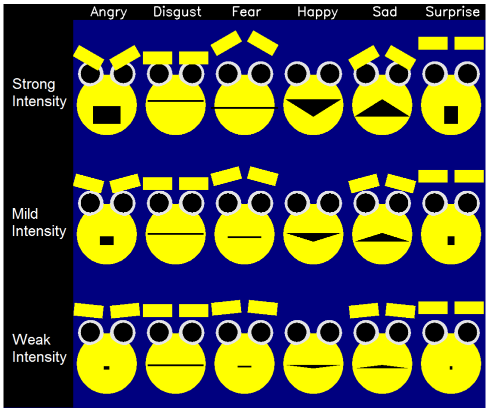


Fig. 23. NPC expressions and intensities (excluding Neutral).

B. Quantitative Evaluation

To test the effectiveness of our system within the application, we performed a quantitative evaluation with the "Art Critic" game. Each player was asked to evaluate paintings using every combination of (non-Neutral) facial expression and verbal evaluation in sequence; this was done 3 times with each player. In all cases, the player had looked at the paintings "long enough" (that is, the Dismissive evaluation is not tested here, since it overlaps with all of the negative evaluations). Six players were tested, giving us a total of 432 samples (= 6 expressions × 4 words × 3 rounds × 6 players).

The results in Table VI demonstrate the effectiveness of our facial expression recognition component in a live context. AUC scores were computed as described earlier. The corresponding confusion matrix is given in Table VII; note that,

[1] Please note that these personalities are not meant to reflect the personalities of the actual artists of the paintings used in this game (e.g., we are not suggesting Jackson Pollock had a Miserable personality nor that Claude Monet had a Cowardly one).

although the players were not instructed to perform Neutral explicitly, it is included here to show cases wherein a given expression was mistaken for Neutral.

TABLE VI
”ART CRITIC” FACIAL EXPRESSION CLASSIFICATION RESULTS WITH HIGHEST ACCURACY (HISTORY SIZE = 50 FRAMES)

Class	Acc.	Pre.	Rec.	F1	AUC
Angry	0.986	0.986	0.986	0.986	0.990
Disgust	0.972	0.959	0.972	0.966	0.993
Fear	0.917	0.985	0.917	0.950	0.976
Happy	1	0.986	1	0.993	0.982
Sad	0.944	0.986	0.944	0.965	0.985
Surprise	0.931	0.957	0.931	0.944	0.995
W. Avg.	**0.958**	**0.977**	**0.958**	**0.967**	**0.987**

TABLE VII
”ART CRITIC” FACIAL EXPRESSION CONFUSION TABLE WITH HIGHEST ACCURACY (HISTORY SIZE = 50 FRAMES)

Classified as →	Angry	Disgust	Fear	Happy	Sad	Surprise	Neutral
Angry	71	1	0	0	0	0	0
Disgust	0	70	0	0	0	0	2
Fear	0	2	66	0	1	1	2
Happy	0	0	0	72	0	0	0
Sad	1	0	0	0	68	2	1
Surprise	0	0	1	1	0	67	3
Neutral	0	0	0	0	0	0	0

The results in Table VIII show how well the system is able to identify the correct painting evaluations given by the player. The corresponding confusion matrix is given in Table IX. Since the speech recognition component worked flawlessly in our tests, we used the weights from the facial expression component to compute AUC. In cases wherein a single painting evaluation could have been generated by multiple expressions, the scores for all relevant expressions are added together for the given evaluation class.

TABLE VIII
”ART CRITIC” PAINTING EVALUATION CLASSIFICATION RESULTS WITH HIGHEST ACCURACY (HISTORY SIZE = 52 FRAMES)

Class	Acc.	Pre.	Rec.	F1	AUC
(Moderate) Dislike	1	0.947	1	0.973	0.996
(Strong) Dislike	1	0.947	1	0.973	0.995
(Moderate) Mocking	1	1	1	1	0.995
(Strong) Mocking	1	1	1	1	0.997
(Moderate) Polite Dislike	0.944	0.971	0.944	0.958	0.994
(Strong) Polite Dislike	0.944	1	0.944	0.971	0.994
(Moderate) Shock	1	1	1	1	1
(Strong) Shock	0.944	0.944	0.944	0.944	0.997
(Moderate) Like	0.944	0.962	0.944	0.953	0.979
(Strong) Like	1	0.964	1	0.982	0.992
(Moderate) Envy	1	0.973	1	0.986	0.995
(Strong) Envy	1	1	1	1	0.996
(Moderate) Awe	0.889	0.941	0.889	0.914	0.999
(Strong) Awe	0.889	1	0.889	0.941	0.999
W. Avg.	**0.972**	**0.973**	**0.972**	**0.972**	**0.993**

Both Tables VI and VIII are from using the best history sizes in terms of classification accuracy (50 and 52 frames, respectively). Fig. 24(a) and Fig. 24(b) show how the accuracy and AUC scores for facial expression classification vary with the history size, while Fig. 24(c) and Fig. 24(d) show how the accuracy and AUC scores for evaluation classification vary with the history size. Both the expression and evaluation AUC scores begin to decline after size 46 or so. A possible explanation is that the history is going too far back in time to a point before the player began making the expression; this would indicate that some of the players may have only held their expression for about a second or so before issuing their evaluation.

TABLE X
”ART CRITIC” QUALITATIVE EVALUATION (SCORES RANGE FROM 1 = “STRONGLY DISAGREE” TO 5 = “STRONGLY AGREE”)

Question	**Mean**	**Std. Dev.**
Overall - Fun	4.50	0.76
Overall - Comfortable	4.33	0.94
Overall - Easy	4.50	0.50
Eye Tracking - Intuitive	4.50	0.50
Eye Tracking - Comfortable	4.00	1.16
Eye Tracking - Immersive	4.33	1.11
Head Pose - Intuitive	5.00	0.00
Head Pose - Comfortable	4.83	0.37
Head Pose - Immersive	4.83	0.37
Facial Expression - Intuitive	4.33	0.75
Facial Expression - Comfortable	4.00	0.82
Facial Expression - Immersive	4.5	0.76
NPC Facial Expression - Immersive	4.33	1.11
NPC Facial Expression - Fun	4.33	1.11
NPC Interaction - Fun	4.50	0.76
NPC Interaction - Immersive	4.50	0.76

C. *Qualitative Evaluation*

After each player finished playing the game, we asked them to fill out a questionnaire about the experience. The questions focused on whether each component, such as head pose, made the game comfortable, more immersive, and/or fun. A 5-point scale was used: “Strongly Disagree” (1), “Disagree” (2), “Neutral” (3), “Agree” (4), and “Strongly Agree” (5). An option for “Not Sure” was also included, but it was not used by any of the players. Table X gives the average and standard deviation of the answers from the questionnaires.

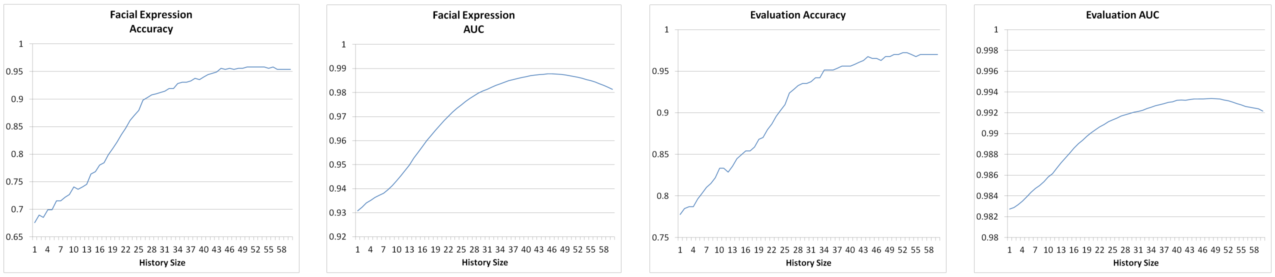
The evaluation shows very positive feedback on the system developed. All the components have an average of at least 4, and most of them meet or exceed 4.5, which demonstrates the positive experience the system generated.

VII. DISCUSSION AND CONCLUSION

In this paper, we have proposed a novel system incorporating head pose estimation, eye gaze estimation, facial expression recognition, speech recognition, and text-to-speech for use in a gaming context. Through the presented game application, we have also shown the utility of these multiple modalities as means of control for more advanced NPC and object interaction as well as, ultimately, increased immersion in a game. The system runs in real time. It is also flexible, able to run with a simple webcam-monitor setup or with a more complex arrangement using a pan-tilt-zoom camera in conjunction with the Kinect. We have also presented a novel head pose estimation algorithm using scene flow and a generic 3D head model, and finally we have shown improved facial

TABLE IX
”ART CRITIC” PAINTING EVALUATION CONFUSION TABLE WITH HIGHEST ACCURACY (HISTORY SIZE = 52 FRAMES)

Classified as →	(Moderate) Dislike	(Strong) Dislike	(Moderate) Mocking	(Strong) Mocking	(Moderate) Polite Dislike	(Strong) Polite Dislike	(Moderate) Shock	(Strong) Shock	(Moderate) Like	(Strong) Like	(Moderate) Envy	(Strong) Envy	(Moderate) Awe	(Strong) Awe
(Moderate) Dislike	36	0	0	0	0	0	0	0	0	0	0	0	0	0
(Strong) Dislike	0	36	0	0	0	0	0	0	0	0	0	0	0	0
(Moderate) Mocking	0	0	18	0	0	0	0	0	0	0	0	0	0	0
(Strong) Mocking	0	0	0	18	0	0	0	0	0	0	0	0	0	0
(Moderate) Polite Dislike	2	0	0	0	34	0	0	0	0	0	0	0	0	0
(Strong) Polite Dislike	0	1	0	0	0	34	0	1	0	0	0	0	0	0
(Moderate) Shock	0	0	0	0	0	0	18	0	0	0	0	0	0	0
(Strong) Shock	0	1	0	0	0	0	0	17	0	0	0	0	0	0
(Moderate) Like	0	0	0	0	1	0	0	0	51	0	1	0	1	0
(Strong) Like	0	0	0	0	0	0	0	0	0	54	0	0	0	0
(Moderate) Envy	0	0	0	0	0	0	0	0	0	0	36	0	0	0
(Strong) Envy	0	0	0	0	0	0	0	0	0	0	0	36	0	0
(Moderate) Awe	0	0	0	0	0	0	0	0	2	0	0	0	16	0
(Strong) Awe	0	0	0	0	0	0	0	0	0	2	0	0	0	16



(a) (b) (c) (d)

Fig. 24. History size vs. (a) facial expression accuracy, (b) facial expression AUC, (c) painting evaluation accuracy, and (d) painting evaluation AUC.

expression recognition performance using LBP-TOP on the 2D shape index image domain.

A word may be said about the computational complexity of the system. The computational complexity for the eye gaze estimation is dominated by solving the linear equations; since SVD is used as the underlying mechanism, the complexity is $O(mn^2)$, where m and n are the number of rows and columns in the matrix, respectively. However, n is constant at 3, so the final complexity for eye gaze estimation is $O(m)$. The facial expression component first involves the calculation of the shape index images. For M pixels and a neighborhood size of N, the resulting complexity is $O(MN^2)$, since the necessary SVD matrices are precomputed ahead of time and the eigenvector/eigenvalue computation for the 2×2 Weingarten matrix takes constant time. We compute uniform LBP features for each new image, so the time complexity is $O(PXY)$, where X and Y are the dimensions of the image and P is the LBP sample count. The LBP features for the new temporal plane data are also computed, so the complexity is $O(B_xPY + B_yPX)$, where B_x and B_y are the number of blocks in x and y, respectively. Finally, the head pose component is composed of two complexity factors. The first is the AAM used for tracking the points; after training, this element runs in linear time to the number of points Q. The second is the scene flow computation, which is also linear in the number of feature points. Thus, the total time complexity for the entire system is approximately $O(m)+O(MN^2)+O(PXY)+O(B_xPY+B_yPX)+O(Q)$, where Q is the number of points in the head pose model.

The current version of our system does have a few limitations. The eye gaze component works best when the accurate orientation and position of the camera and screen is known ahead of time (i.e., the camera system is “fully calibrated”). Still, an estimate of the eye gaze focus point can still be obtained without precise screen-camera information. Our current expression module uses person-specific templates; however, it is not unusually for computer-vision-based games to require a person-specific calibration phase. Nonetheless, in the future we will collect and train on a large facial expression database to provide a more flexible, reliable, and universal solution. Finally, the head pose estimation approach does rely on the accuracy of the tracked feature points, and therefore we will work on increasing the robustness of the feature point tracking.

Overall, the underlying interaction system of the “Art Critic” game has great potential to advance the state of the art in the gaming arena and more generally in human-computer interaction. Immersion and suspension of disbelief have remained elusive goals in game development, partially due the hitherto unavoidable but nevertheless clumsy ways the player interfaces with the game world and its virtual inhabitants (e.g., selecting among "verbal" responses with a mouse, manually adjusting the expression of the player's avatar, etc.). With our system, however, the combination of head pose, eye gaze, expression, speech information, and synthetic speech forms a more complete communication interface, one that feels intuitive and behaves naturally. A single-channel system is insufficient, as it would lead to unnatural and uncomfortable control mapping schemes. For example, how does one map facial expression to player view? The problem remains even if the mapping scheme is reasonable; for instance, if eye gaze alone was used to change the viewing direction, the player would have to maintain perfect control of his/her eyes, fixing steadily on a location, which is awkward at best. Even using a subset of these channels would run the risk of making the technology a mere gimmick, since other, more conventional means of communication would have to fill the gap and thus remind the player that all this is, indeed, only a game. Another advantage of using multiple channels is that it allows the system to recognize more complex and interesting responses from the player, such as Mocking (which combines two seemingly conflicting signals, a smile and a negative verbal evaluation).

Moreover, we would argue that the use of player information in "Art Critic" is sensible and logical, and we believe that the concepts presented herein could be easily extended to other gaming scenarios. For example, imagine a fighting game wherein the player can taunt or intimidate his/her opponent using speech, gaze, head gestures, and facial expression. Alternatively, one can envisage a first-person shooter game wherein the player encourages surrounding NPC troops and/or issues orders using the same channels of communication. Another possible scenario would be a simulated social interaction game for children. Indeed, any virtual world that involves one or more non-human agents can only feel immersive if the agents respond naturally to all of the signals the player is sending, and our system enables virtual agents to do so. If we apply these concepts to "edutainment", educational games could be made more engaging and effective if the system is able to recognize the child's speech and interpret his/her facial expressions, gaze, and head/hand gestures, e.g., to see whether the child interested, bored, or confused.

Our future work in this area will include increasing the robustness of each system component. We would also like to extend our expression recognition system in the following ways: 1) use the eye gaze as a cue to integrate with dynamic textures for estimation of emotion and intention; 2) expand its list of expressions to non-standard affective states, such as interest or boredom; 3) train it to be able to recognize subtle or low-intensity facial expressions; and 4) extend it to output the expression intensity. We would also be interested in including head gestures (e.g., nodding "yes" or shaking the head "no") as part of the overall system. One of our goals is utilizing hand gestures or body movement/position (such as from the Kinect) as indicators to move forward and backward in the virtual world [45] [46]. In general, further integration of the Kinect into our system is another of our project goals, perhaps making direct use of the depth imagery. More advanced NPC artificial intelligence would also be of interest. Finally, we would like the system to interpret more complicated speech instructions as well as to infer emotions from speech amplitude and intonation [47].

References

[1] D. Rowan, "Kinect for Xbox 360: The inside story of Microsoft's secret 'Project Natal'," *Wired Magazine*, 2010.

[2] "PlayStation®Move website," 2012. [Online]. Available: http://us.playstation.com/ps3/playstation-move/

[3] M. Picheny, D. Nahamoo, V. Goel, B. Kingsbury, B. Ramabhadran, S. J. Rennie, and G. Saon, "Trends and advances in speech recognition," *IBM Journal of Research and Development*, vol. 55, no. 5, pp. 2:1–2:18, Sept.-Oct. 2011.

[4] M. E. Ayadi, M. S. Kamel, and F. Karray, "Survey on speech emotion recognition: Features, classification schemes, and databases," *Pattern Recognition*, vol. 44, no. 3, pp. 572–587, 2011.

[5] D. Hansen and Q. Ji, "In the eye of the beholder: a survey of models for eyes and gaze," *IEEE Trans. on PAMI*, vol. 32, pp. 478–500, 2010.

[6] D. Xia and Z. Ruan, "IR image based eye gaze estimation," in *ACIS Intl. Conf. Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing (SNPD'07)*, vol. 1, 2007, pp. 220–224.

[7] C. Colombo, D. Comanducci, and A. D. Bimbo, "Robust tracking and remapping of eye appearance with passive computer vision," *ACM Trans. on Multimedia Computing, Communications, and Applications*, vol. 3, pp. 2:1–2:20, Dec. 2007.

[8] J. Wang, L. Yin, and J. Moore, "Using geometric properties of topographic manifold to detect and track eyes for human-computer interaction," *ACM Trans. on Multimedia Computing, Communications, and Applications*, vol. 3, pp. 3:1–3:20, Dec. 2007.

[9] E. Pogalin, A. Redert, I. Patras, and E. Hendriks, "Gaze tracking by using factorized likelihoods particle filtering and stereo vision," in *Intl. Symp. 3D Data Processing, Visualization, and Transmission (3DPVT'06)*, 2006, pp. 57–64.

[10] J. Xie and X. Lin, "Gaze direction estimation based on natural head movements," in *Intern. Conf. on Image and Graphics (ICIG'07)*, 2007, pp. 672–677.

[11] H. Wu, Y. Kitagawa, T. Wada, T. Kato, and Q. Chen, "Tracking iris contour with a 3d eye-model for gaze estimation," in *Asian Conf. on Computer Vision - Volume Part I (ACCV'07)*. Springer-Verlag, 2007, pp. 688–697.

[12] H. Yamazoe, A. Utsumi, T. Yonezawa, and S. Abe, "Remote and head-motion-free gaze tracking for real environments with automated head-eye model calibrations," in *IEEE CVPR Workshop for Human Communicative Behavior Analysis (CVPR4HB)*, 2008, pp. 1–6.

[13] D. Schnieders, X. Fu, and K.-Y. Wong, "Reconstruction of display and eyes from a single image," in *IEEE Conf. on Computer Vison and Pattern Recognition (CVPR'10)*, Jun. 2010, pp. 1442–1449.

[14] M. Breitenstein, D. Kuettel, T. Weise, L. van Gool, and H. Pfister, "Real-time face pose estimation from single range images," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR'08)*, Jun. 2008, pp. 1–8.

[15] G. Fanelli, T. Weise, J. Gall, and L. V. Gool, "Real time head pose estimation from consumer depth cameras," in *33rd Intern. Conf. on Pattern recognition*, ser. DAGM'11. Berlin, Heidelberg: Springer-Verlag, 2011, pp. 101–110.

[16] M. Fischler and R. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," *Communications of the ACM*, vol. 24, pp. 381–395, Jun. 1981.

[17] Y. Tong, Y. Wang, Z. Zhu, and Q. Ji, "Robust facial feature tracking under varying face pose and facial expression," *Pattern Recognition*, vol. 40, pp. 3195–3208, Nov. 2007.

[18] C. Zhan, W. Li, P. Ogunbona, and F. Safaei, "A real-time facial expression recognition system for online games," *Intern. Journal of Computer Games Technology - Joint Intern. Conf. on Cyber Games and Interactive Entertainment*, vol. 2008, pp. 10:1–10:7, Jan. 2008.

[19] X. Zhou, X. Huang, and Y. Wang, “Real-time facial expression recognition in the interactive game based on embedded hidden markov model,” in *Intern. Conf. on Computer Graphics, Imaging and Visualization (CGIV’04)*, Jul. 2004, pp. 144–148.

[20] N. Magnenat-Thalmann and Z. Kasap, “Virtual humans in serious games,” in *Intl. Conf. on CyberWorlds (CW’09)*, Sept. 2009, pp. 71–79.

[21] W.-P. Su, B. Pham, and A. Wardhani, “Personality and emotion-based high-level control of affective story characters,” *IEEE Trans. on Visualization and Computer Graphics*, vol. 13, pp. 281–293, Mar. 2007.

[22] S.-M. Choi and Y.-G. Kim, “An affective user interface based on facial expression recognition and eye-gaze tracking,” in *Affective Computing and Intelligent Interaction*, ser. Lecture Notes in Computer Science. Springer-Verlag, 2005.

[23] J. Cassell, T. Bickmore, M. Billinghurst, L. Campbell, K. Chang, H. Vilhjálmsson, and H. Yan, “An architecture for embodied conversational characters,” in *Proceedings of Computer Animation and Simulation.* Springer-Verlag, 1998, pp. 109–120.

[24] H. Gómez-Gauchía and F. Peinado, “Automatic customization of non-player characters using players temperament,” in *Technologies for Interactive Digital Storytelling and Entertainment*, ser. Lecture Notes in Computer Science. Springer-Verlag, 2006.

[25] C. Kozasa, H. Fukutake, H. Notsu, Y. Okada, and K. Niijima, “Facial animation using emotional model,” in *Intern. Conf. on Computer Graphics, Imaging and Visualisation*, Jul. 2006, pp. 428–433.

[26] B. Lance and S. Marsella, “A model of gaze for the purpose of emotional expression in virtual embodied agents,” in *Intern. Joint Conf. on Autonomous Agents and Multiagent Systems*, vol. 1.

[27] M. Reale, S. Canavan, L. Yin, K. Hu, and T. Hung, “A multi-gesture interaction system using a 3-D iris disk model for gaze estimation and an active appearance model for 3-D hand pointing,” *IEEE Trans. on Multimedia*, vol. 13, no. 3, pp. 474–486, Jun. 2011.

[28] S. Vedula, P. Rander, R. Collins, and T. Kanade, “Three-dimensional scene flow,” *IEEE Trans. on PAMI*, vol. 27, no. 3, pp. 475–480, Mar. 2005.

[29] T. F. Cootes, G. J. Edwards, and C. J. Taylor, “Active appearance models.” *IEEE Trans. on PAMI*, vol. 23, no. 6, pp. 681–685, 2001.

[30] C. Dorai and A. Jain, “COSMOS-a representation scheme for 3D free-form objects,” *IEEE Trans. on PAMI*, vol. 19, no. 10, pp. 1115 –1130, Oct. 1997.

[31] G. Zhao and M. Pietikäinen, “Dynamic texture recognition using local binary patterns with an application to facial expressions,” *IEEE Trans. on PAMI*, vol. 29, no. 6, pp. 915–928, Jun. 2007.

[32] “cmusphinx.org,” *CMU Sphinx: The Carnegie Mellon Sphinx Project*, 2012. [Online]. Available: http://cmusphinx.org

[33] V. Strom and S. King, “Investigating Festival’s target cost function using perceptual experiments,” in *INTERSPEECH, 9th Annual Conf. of the Intern. Speech Communication Association*, 2008.

[34] P. Viola and M. J. Jones, “Robust real-time face detection,” *Intern. Journal of Computer Vision*, vol. 57, no. 2, pp. 137–154, May 2004.

[35] P. Martins and J. Batista, “Accurate single view model-based head pose estimation,” in *IEEE Intern. Conf. on Automatic Face & Gesture Recognition (FG’08)*, Sept. 2008, pp. 1–6.

[36] X. Zhang, L. Yin, J. Cohn, S. Canavan, M. Reale, A. Horowitz, and P. Liu, “A high-resolution spontaneous 3D dynamic facial expression database,” *IEEE Intern. Conf. on Automatic Face & Gesture Recognition (FG’13)*, Apr. 2013.

[37] “Di3D. Inc.” 2012. [Online]. Available: http://www.di3d.com

[38] “NVIDIA CUDA website,” 2012. [Online]. Available: http://www.nvidia.com/object/cuda_home_new.html

[39] D. Li, D. Winfield, and D. Parkhurst, “Starburst: A hybrid algorithm for video-based eye tracking combining feature-based and model-based approaches,” in *IEEE CVPR Workshop on Vision for Human-Computer Interaction (V4HCI)*, Jun. 2005, pp. 79–79.

[40] “GLSL website,” 2012. [Online]. Available: http://www.opengl.org/documentation/glsl/

[41] J. Wang and L. Yin, “Static topographic modeling for facial expression recognition and analysis,” *Computer Vision and Image Understanding*, vol. 108, no. 1-2, pp. 19–34, Oct. 2007.

[42] J. Goldfeather and V. Interrante, “A novel cubic-order algorithm for approximating principal direction vectors,” *ACM Trans. on Graphics*, vol. 23, pp. 45–63, Jan. 2004.

[43] L. Yin, X. Chen, Y. Sun, T. Worm, and M. Reale, “A high-resolution 3D dynamic facial expression database,” in *IEEE Intern. Conf. on Automatic Face & Gesture Recognition (FG’08)*, Sept. 2008, pp. 1–6.

[44] G. Tzanetakis, *ICME 2004 Tutorial : Audio Feature Extraction*, 2004. [Online]. Available: webhome.cs.uvic.ca/ gtzan/work/talks/icme04/icme04tutorial.pdf

[45] N. D. Sonwane, S. Chhabria, and R. Dharaskar, “Speech and gesture recognition using som with markov model,” *Intern. Journal of Research in Image, Video and Signal Processing (IJRIVSP)*, vol. 1, 2012. [Online]. Available: http://googlejournals.in/GJ/index.php/IJRIVSP/article/view/66

[46] J. Vascellaro, “Hand-gesture technologies wave bye to desktop mouse,” *Wall Street Journal Technology*, 2012. [Online]. Available: http://online.wsj.com/article/SB10001424052702303879604577412550939856014.html

[47] S. G. Koolagudi and K. S. Rao, “Emotion recognition from speech: a review,” *Intern. Journal of Speech Technology*, vol. 15, pp. 99–117, 2012. [Online]. Available: http://dx.doi.org/10.1007/s10772-011-9125-1

Michael J. Reale (S’10) graduated with his B.S. in Computer Science from SUNY Oneonta, New York, in 2007. In May 2009, he received his M.S. from the State University of New York at Binghamton, and he is now a Ph.D. candidate at the same working in the Graphics and Image Computing (GAIC) lab. His research interests include eye tracking and gaze estimation, human-computer interaction interfaces, expression recognition, and computer graphics, as well as GPGPU programming for computer vision.

Peng Liu (S’12) received the B.S. degree in Electronic Information Science and Technology from University of Science and Technology of China, Hefei, China in 2006 and received the M.S. degree in Biomedical Engineering from University of Science and Technology of China, Hefei, China in 2009. He is now pursuing the Ph.D. degree in the Graphics and Image Computing (GAIC) lab at the State University of New York at Binghamton. His research interests are head pose estimation, face recognition, human-computer interaction, and computer graphics.

Lijun Yin (M’00 - SM’10) received the MSc degree in Electrical Engineering from Shanghai Jiao Tong University in 1992 and the Ph.D. degree in Computer Science from the University of Alberta, Canada in 2000. He joined the State University of New York at Binghamton in 2001. He is currently an associate professor and the director of the Graphics and Image Computing (GAIC) Laboratory in the Computer Science Department, SUNY at Binghamton. He was a summer visiting faculty in the Air Force Research Lab at Rome, NY in 2005. His research interests include visual information processing, computer vision, biometrics, face and gesture analysis, recognition, animation, and applications to human computer interaction. He has over 100 publications in referred journals and conferences. His research has been supported by the National Science Foundation, the New York State Office of Science, Technology and Academic Research (NYSTAR), Air Force Research Lab, and the SUNY Upstate Medical Center. Dr. Yin received the prestigious NYSTAR’s James Watson Young Investigator Award in 2006. He is currently serving as an Editorial Board Member for two Journals (Image and Vision Computing; Pattern Recognition Letters). He is also a program chair of the 10th IEEE International Conference on Automatic Face and Gesture Recognition (FG 2013).

Shaun Canavan (S’07) received the B.S. degree in Computer Science and the MCIS degree in Computer Information Systems from Youngstown State University in 2006 and 2008, respectively. He is currently a Ph.D. candidate in Computer Science with the Graphics and Image Computing (GAIC) lab at the State University of New York at Binghamton. Shaun was selected to study at the SINO-USA summer school in vision, learning, and pattern recognition in Xi’an China (2010). He was also selected for the International Joint Conference on Biometrics doctoral consortium in 2011. His research interests include statistical shape analysis of 2D and 3D shapes, face detection and recognition, human-computer interaction, biometrics, and computer graphics research.