Head Pose Estimation and Augmented Reality Tracking: An Integrated System and Evaluation for Monitoring Driver Awareness

Erik Murphy-Chutorian, *Member, IEEE*, and Mohan Manubhai Trivedi, *Fellow, IEEE*

***Abstract*—Driver distraction and inattention are prominent causes of automotive collisions. To enable driver-assistance systems to address these problems, we require new sensing approaches to infer a driver's focus of attention. In this paper, we present a new procedure for static head-pose estimation and a new algorithm for visual 3-D tracking. They are integrated into the novel real-time (30 fps) system for measuring the position and orientation of a driver's head. This system consists of three interconnected modules that detect the driver's head, provide initial estimates of the head's pose, and continuously track its position and orientation in six degrees of freedom. The head-detection module consists of an array of Haar-wavelet Adaboost cascades. The initial pose estimation module employs localized gradient orientation (LGO) histograms as input to support vector regressors (SVRs). The tracking module provides a fine estimate of the 3-D motion of the head using a new appearance-based particle filter for 3-D model tracking in an augmented reality environment. We describe our implementation that utilizes OpenGL-optimized graphics hardware to efficiently compute particle samples in real time. To demonstrate the suitability of this system for real driving situations, we provide a comprehensive evaluation with drivers of varying ages, race, and sex spanning daytime and nighttime conditions. To quantitatively measure the accuracy of system, we compare its estimation results to a marker-based cinematic motion-capture system installed in the automotive testbed.**

***Index Terms*—Active safety, graphics programming units, head pose estimation, human-computer interface, intelligent driver assistance, performance metrics and evaluation, real-time machine vision, support vector classifiers, 3-D face models and tracking.**

I. Introduction

VEHICULAR safety relies on the ability of people to maintain constant awareness of the environment as they drive. As new vehicles and obstacles move into the vicinity of the car, a driver must be cognizant of the change and be ready to respond as necessary. Although people have an astounding ability to cope with these changes, a driver is fundamentally limited by the field of view that he can observe at any one time. When a driver fails to notice a change to his environment, there is an increased potential for a life-threatening collision. It is reasonable to assume that this danger could be mitigated if the driver is notified when these situations arise. As evidence to this effect, a recent comprehensive survey on automotive collisions demonstrated that a driver was 31% less likely to cause an injury-related collision when he had one or more passengers who could alert him to unseen hazards [1]. Consequently, there is great potential for driver-assistance systems that act as virtual passengers, alerting the driver to potential dangers through aural or visual cues [2]. To design such a system in a manner that is neither distracting nor bothersome, these systems must act like real passengers, alerting the driver only in situations where he appears to be unaware of the possible hazard. This requires a context-aware system that simultaneously monitors the environment and actively interprets the behavior of the driver. By fusing information from inside and outside the vehicle, automotive systems can better model the circumstances that motivate driver behavior [3], [4].

With consideration for future driver assistance systems, we concentrate on one of the integral processes for monitoring driver awareness: estimation of the position and orientation of a driver's head. Head pose is a strong indicator of a driver's field of view and current focus of attention. It is intrinsically linked to visual *gaze estimation*, which is the ability to characterize the direction in which a person is looking [5], [6]. Intuitively, it might seem that looking at the driver's eyes might provide a better estimate of gaze direction, but in the case of lane-change intent prediction, for example, head dynamics were shown to be a more reliable cue [7]. In addition, implementing a vision system that focuses on a driver's eyes is impractical at many levels. In addition to the economic and technical challenges of integrating and calibrating multiple high-resolution cameras placed throughout the cabin (to view the eye from all head positions), it requires that the driver's eyes be visible at all times (e.g., sunglasses or other eye-occluding objects would cause the system to malfunction). Furthermore, we believe that the eyes can convey only the gaze direction relative to the direction of the head. Physiological studies demonstrate that this is clearly the case for human perception [8], and computational eye trackers typically require the subject to maintain a frontal head pose.

Manuscript received September 22, 2007; revised January 19, 2008, July 20, 2009, November 18, 2009, and January 6, 2010; accepted January 15, 2010. Date of publication April 5, 2010; date of current version May 25, 2010. The Associate Editor for this paper was Q. Ji.

E. Murphy-Chutorian was with the Computer Vision and Robotics Research Laboratory, Department of Electrical and Computer Engineering, University of California, San Diego, La Jolla, CA 92093 USA. He is now with Google Inc., Mountain View, CA 94043 USA (e-mail: erikmc@ucsd.edu; erikmc@google.com).

M. M. Trivedi is with the Computer Vision and Robotics Research Laboratory, Department of Electrical and Computer Engineering, University of California, San Diego, La Jolla, CA 92093 USA (e-mail: mtrivedi@ucsd.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TITS.2010.2044241

1524-9050/$26.00 © 2010 IEEE

Computational head pose estimation remains a challenging vision problem, and there are no solutions that are both inexpensive and widely available. Among the research thrusts and commercial offerings that can provide a real-time estimate of head pose, most require multiple cameras to obtain a correspondence-based depth information, and none have been rigorously and quantitatively evaluated in an automobile. In a car, ever-shifting lighting conditions cause heavy shadows and illumination changes, and as a result, techniques that demonstrate high proficiency in stable lighting often will not work in these situations.

The novelty of this paper is threefold: First, we introduce a new procedure for head pose estimation and a new algorithm for 3-D head tracking. Second, we provide a systematic implementation of these two to create a hybrid head-pose estimation system. In this computational system, we only use a single video camera and provide a real-time (30 fps) implementation by optimizing the calculations for the parallel processors available on a consumer graphic processor. Third, we quantitatively demonstrate the success of this system on the road, comparing our markerless monocular head-pose estimator to ground truth obtained with a professional cinematic motion-capture system that we have configured for a vehicular testbed. To ensure a wide variety of driving conditions, we perform these experiments with drivers of varying age, race, and sex spanning daytime and nighttime drives.

In designing our system, we strove for a cost-efficient prototype that could be reasonably adapted for widely deployed automobiles. Although our prototype has been implemented in a full-size PC, the current evolution of embedded processors (and embedded graphics processors) would be the natural progression for the future of this technology. Our system was designed to meet the following design criteria:

1) Monocular: The system must be able to estimate head pose from a single camera. Although accuracy might be improved with stereo imagery, multiple cameras increase the cost and complexity of the system, and they require manual calibration that can drift as a result of vibrations and impacts.
2) Autonomous: There should be no manual initialization, and the system should operate without any human intervention. This criterion precludes the use of pure-tracking approaches that measure the relative head pose with respect to some initial configuration.
3) Fast: The system must be able to estimate a continuous range of head pose while driving, with real-time (30 fps) operation.
4) Identity and Lighting Invariant: The system must work across different drivers in varying lighting conditions.

II. Prior Work

Recently, there has been a great interest in driver-assistance systems that use computer vision technology to develop safer automobiles [3]. Within this scope, a large area of focus has been to direct cameras inside the vehicle and interpret the driver's state from video observations.

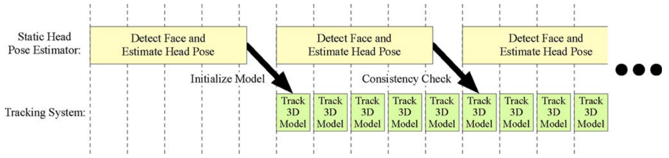

Static Head Pose Estimator:
Detect Face and Estimate Head Pose
Detect Face and Estimate Head Pose
Detect Face and Estimate Head Pose
Initialize Model
Consistency Check
Tracking System:
Track 3D Model

Fig. 1. Our hybrid head-pose-estimation scheme combines a static head-pose estimator with a real-time 3-D model-based tracking system. The static estimator initializes the tracker and provides periodic consistency checks as the two operations run in parallel.

In one prime example, the driver's eye closure blink frequency, nodding frequency, and 2-D face position have been used to estimate driver attentiveness [9]. This system uses infrared illuminators and Kalman filters to track the driver's pupil and a fuzzy classifier to provide an overall estimate of attentiveness. In another system, the driver's eyes and lip corners were initialized with color predicates and tracked in relation to a bounding box around the driver's head [10]. This was shown to provide an estimate of the driver's gaze as well as to estimate the driver attentiveness level using finite-state machines.

Driver head-motion estimation has also been used along with video-based lane detection and CAN bus data to predict the driver's intent to change lanes in advance of the actual movement of the vehicle [11]. This paper supplied these cues to a sparse Bayesian learning classifier that provides a probabilistic prediction of a lane change seconds in advance.

All these previous works use a coarse estimate of head motion as the input to a classifier that estimates an aspect of the driver's intent. In contrast, our system provides a fine absolute measure of the driver's head position that can directly be used to indicate the driver's focus of attention. As a result, we have put great attention into ensuring that the system is robust to variations in lighting and driver's appearance, and we have evaluated the accuracy of this system in varying conditions.

Our contributions in this paper also include new algorithms for head pose estimation and tracking, and we present a review of prior works in this area.

Our system is a hybrid approach that combines the initialization and stability properties of a static pose estimator with the highly accurate, jitter-free, and real-time capabilities of a tracking approach. The static estimator initializes the tracker from a single image frame and, as the head is tracked, continues to run in parallel, providing a periodic consistency check. If the tracking confidence falls below a threshold or the consistency check fails, then the static estimator automatically reinitializes the tracker. This process is illustrated in Fig. 1. Although very different in composition and scope, other works have espoused the advantages of hybrid systems [12]–[15].

The static head-pose estimator that we have developed is a nonlinear regression technique that directly estimates the head pose from a detected image patch. Nonlinear regression approaches provide continuous estimates of pose and have the some of the highest reported success in indoor environments [12]. Prior work in this area includes locally linear maps [16], multilayer perceptrons [17], and principal component analysis (PCA) projection with support vector regression [18]. From our experience with nonlinear regressors, we have observed that the most significant problem with these approaches is their sensitivity to localization error. With noisy face localization (as is common with computational face detectors), the accuracy of these approaches diminishes. In our investigations, we found that we can mitigate this problem by using localized gradient orientation (LGO) histograms as the input to nonlinear regressors. These histograms provide explicit invariance to face localization error, as well as added invariance to lighting and appearance variation. In this paper, we provide experimental evaluation of the improvement in pose estimation by extracting these histograms.

Unlike static head pose estimation techniques, head tracking approaches operate on continuous video, estimating head pose by inferring the change in pose between consecutive frames of a video sequence. These approaches exploit the temporal continuity and smooth motion constraints to provide a jitter-free estimate of pose over time. These systems typically demonstrate much higher levels of accuracy than static pose estimation methods, but they require initialization from a known head position and are prone to drifting and losing track. Our system is an example of a top-down tracking approach that finds a global transformation that best accounts for the observed motion between video frames. With stereo imagery, for instance, the head pose can also be obtained with affine transformations by finding the translation and rotation that minimize the discrepancy in grayscale intensity and depth [14]. In addition to finding the transformation that minimizes the appearance between the model and the new camera frame, systems can also incorporate prior information about the dynamics of the head. Particle filters provide an approximation of the optimal track by maximizing the posterior probability of the movement from a simulated set of samples. Variations on particle filtering have been applied to head accurate real-time head-pose tracking in varying environments, including near-field video [19], low-resolution video with adaptive PCA subspaces [20], and near-field stereo with affine approximations [21], [22]. In this paper, we introduce a new dual-state particle filter to explicitly model the nonlinear motion of a driver. This motion model is able to simultaneously account for the observed jitter of a driver's head and the driver's intentional head movements. Compared with other particle tracking approaches, we have overcome many simplifications and limitations such that we have the following.

1) We only require monocular video, satisfying our design criterion and preventing the need for periodic stereo calibration.
2) We compute full projective transformations of the model, rather than affine approximations, improving performance by removing an artificial source of distortion.
3) We use a full textured-mapped 3-D model instead of a series of point samples, allowing a more complete comparison between the model and the observation.
4) We provide a real-time implementation, satisfying our design criterion for 30-fps tracking.

The system that we present in this paper is a novel software engine that advances the state of the art in fully autonomous head-pose estimation. This system has practical utility for many applications including intelligent meeting spaces [23], and in this paper, we focus our efforts on the automotive domain. To demonstrate the capacity of our system, we evaluated it on a wide range of natural driving situations with a cinematic motion-capture system providing a quantitative comparison. Although there have been other head-pose-estimation systems that have been applied to automotive imagery [24]–[29], they have been evaluated when the car is moving in specific scenarios only in situations where the car is not moving (indoors). It is unclear whether these approaches would require substantial modification to become viable options for real automotive use. In contrast, the data collection and evaluation that we have conducted in this paper are the first their kind, and we are able to demonstrate that our system attains a high level of accuracy during real-world driving.

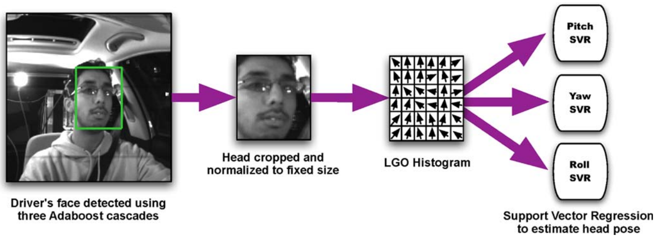

Driver's face detected using three Adaboost cascades
Head cropped and normalized to fixed size
LGO Histogram
Pitch SVR
Yaw SVR
Roll SVR
Support Vector Regression to estimate head pose

Fig. 2. Overview of our static head-pose-estimation procedure consisting of three steps: 1) The head is detected with a trio of cascaded Adaboost detectors. 2) An LGO histogram is extracted from the cropped head region. 3) The histogram is passed to SVRs for pitch, yaw, and roll.

The remainder of this paper is structured as follows: Section III details our methods for head detection and static head-pose estimation. Section IV introduces our augmented-reality head-tracking algorithm. Section V describes our hybrid head pose system and the real-time implementation of the tracker using optimized consumer-grade graphics hardware. Section VI introduces our automotive testbed and presents an evaluation of our methods. Section VII contains our concluding remarks.

III. Face Detection and Head-Pose Estimation

In the first stage of our system, we compute an initial estimate of the driver's head position and orientation. This consists of the following three steps.

1) A facial region is found using three cascaded Adaboost [30] face detectors applied to the grayscale video images.
2) The detected facial region is scale normalized to a fixed size and used to compute an LGO histogram.
3) The histogram is passed to three support vector regressors (SVRs) trained for head pitch, yaw, and roll.

A graphical overview of this procedure is presented in Fig. 2. It is run once to initialize the tracker and periodically repeated to check the consistency of the tracking estimate. In this following paragraphs, we describe these steps in more detail.

A. *Facial Region Detector*

To detect the location of the driver's head, we use three Adaboost cascades attuned to the left profile, frontal, and right profile faces [30], [31]. Each detector is capable of recognizing heads with enough deviation from its characteristic pose that when combined, they span the range of head poses in our

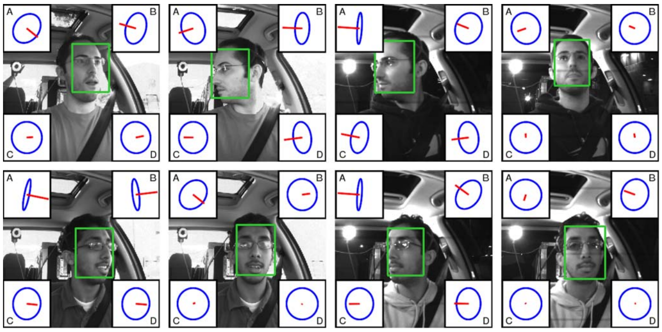


Fig. 3. Comparison of static head-pose estimation using the following methods: (A) NCC prototype matching. (B) Gradient PCA with support vector regression [18]. (C) LGO histograms with support vector regression. (D) Vicon motion capture ground truth. The center rectangle indicates the detected facial region using a trio of cascaded Adaboost face detectors, and the pose for each method is indicated by the direction of the thumbtack.

training data: -30° to 30° in pitch and roll -90° to 90° in yaw. For both training and testing, an uncompressed grayscale image is used as the input to the detectors, and we consider the largest detected rectangular region as the location of the driver's face. To ensure that the static pose estimation process is invariant to scale, every region is down sampled to a fixed size of 34×34 pixels. In an automobile, this makes the system invariant to the distance between the driver and the camera. In our experiments, this facial-detection scheme successfully detected a region in approximately 90% of the video frames. For the remaining frames, the initialization or consistency check is simply skipped until the next successful detection. No effort was made to prune false detections, although one could envision a production system with heuristics based on size, position, and color. From our experience with these detectors in driving video, false detections are quite rare, but when they do occur, the pose estimates are clearly incorrect until the next successful face detection. The detection examples are illustrated in Fig. 3.

B. LGO Histogram

To provide a robust description of each facial region, we compute the LGO histogram. A fixed-size version of this representation was first presented as part of the scale-invariant feature transform [32], which is intended for correspondence matching between regions surrounding scale- and rotation-invariant keypoints. It is a compact feature representation that is robust to minor deviations in region alignment, lighting, and shape [32], [33]. This is useful for automatic head pose estimation, since the explicit position invariance of the histogram offsets some of the localization error from the face detector. Additionally, the histogram is invariant to affine lighting changes, and the gradient operation emphasizes edge contours that are less influenced by identity than image texture. The merit of the generalized histogram has been demonstrated for human detection, where it has alternatively been called a histogram of oriented gradients [34]. In contrast to object recognition systems that represent an object as a configuration of multiple histogram descriptors [32], we use a single LGO histogram to represent the entire scale-normalized facial region. This descriptor consists of a 3-D histogram. The first two dimensions correspond to the vertical and horizontal positions in the image and the third to the gradient orientation. For an $M \times N \times O$ histogram, let the triplet (m, n, o) denote a specific bin in the histogram. The horizontal and vertical image gradients $\boldsymbol{X}_x(x, y)$ and $\boldsymbol{X}_y(x, y)$ are approximated by filtering with 3×3 pixel Sobel kernels. The image is then split into $M \times N$ discrete blocks, and for each pixel (x, y) in the (m, n) block, the absolute gradient orientation $o_{x,y}$ is quantized into one of O discrete levels

$$o_{x,y} = \left\lfloor O \times \left(\frac{1}{2\pi} \operatorname{atan2}\left(\boldsymbol{X}_y(x, y), \boldsymbol{X}_x(x, y)\right) + 0.5 \right) \right\rfloor \qquad (1)$$

and used to increment the $(m, n, o_{x,y})$ histogram bin. After computing the histogram, it is smoothed with the $3 \times 3 \times 3$ kernel as

$$K(m, n, o) = \left(1 - \frac{g(m)}{M}\right)\left(1 - \frac{g(n)}{N}\right)\left(1 - \frac{g(o)}{O}\right) \qquad (2)$$

to prevent aliasing effects, where $\{m, n, o \in \mathcal{B}\}$ for $\mathcal{B} = \{-1, 0, 1\}$, and $g(\cdot)$ is the complement impulse function

$$g(\lambda) = \begin{cases} 0, & \text{if } \lambda = 0 \\ 1, & \text{if } \lambda \neq 0. \end{cases} \qquad (3)$$

The resulting *soft* histogram is subsequently reshaped and normalized to a unit vector. Finally, as suggested by Lowe [32], any vector component greater than 0.2 is truncated to 0.2, and the vector is renormalized if necessary. In our system, we use a 128-D vector, where $M = 4$, $N = 4$, and $O = 8$.

C. Support Vector Regression

To estimate the pose of the driver's head, we use support vector regression on the LGO histogram inputs. Support vector regression is a supervised learning technique for the nonlinear regression of a scalar function [35], [36].

An optimized software package [37] was used to train our system with radial basis function kernels. We generated three regressors trained for head pitch, yaw, and roll. The input to each is the LGO histogram described in Section III-B. To find the optimum regression parameters, we scale normalize each component of the input data such that it spans the range [−1, 1] and then perform a cross-validation grid search across the parameter space. During testing, we use the same scaling parameters to normalize the new input before predicting the new pose.

IV. Head-Pose Tracking in Augmented Reality

We introduce a new procedure to track the driver's head in six degrees of freedom at 30 fps from a single video camera. Our approach uses an appearance-based particle filter in an *augmented reality*, which is a virtual environment that mimics the view space of a real camera [23]. Using an initial estimate of the head position and orientation, the system generates a texture-mapped 3-D model of the head from the most recent video image and places it into the environment. The model is subsequently rotated, translated, and rendered in perspective projection to match the view from each subsequent video frame. It would be computationally inefficient to exhaustively search for the best transformation, so instead, we introduce an appearance-based particle filter framework to generate a set of virtual samples that together provide an optimal estimate of this transformation. The virtual samples are perspective projections of the head model at a specific rotation and translation and resemble small perturbations of the driver's face set against a solid background.

Although the 3-D construction and evaluation of these samples is a daunting computational challenge for a conventional computer processor, we show that it can be highly optimized for graphic processing units (GPUs), and we describe our real-time implementation that utilizes the 3-D virtualization and processing capabilities of a consumer-level GPU.

This section is organized in the following manner. Part A describes our dual-state motion model, and Part B details our particle-filtering approach to update this model.

A. State Model

We represent the driver's head as a rigid object constrained to six degrees of freedom in a 3-D world. This can be represented with respect to a fixed Cartesian coordinate system by the position, (x, y, z) and Euler angles (α, β, γ). To model the system using linear dynamics, we could define the state

$$\boldsymbol{x}_t = \begin{bmatrix} \boldsymbol{\theta_t} \\ \boldsymbol{\omega_t} \end{bmatrix} \tag{4}$$

where $\boldsymbol{\theta}_t = [x_t, y_t, z_t, \alpha_t, \beta_t, \gamma_t]^T$ represents the position and angle of the object at time t, and $\boldsymbol{\omega}_t$ represents the respective linear and angular velocity. In our head tracking application, however, motion is not well described by a linear system. Consider the typical motion of a person's head bobbling about in an automobile. For the most part, the subject is focused on a single location in the world, and his head is essentially static, subject only to small perturbations that can appear to be instantaneous when viewed at a sampling rate defined by a video camera. Only when the person conscientiously moves their head from one position to another can linear dynamics provide a good temporary approximation of the motion. The first situation can be modeled with a zero-velocity state model

$$\boldsymbol{x}_t^{(ZV)} = \begin{bmatrix} \mathbf{1} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \boldsymbol{x}_{t-1} + \begin{bmatrix} \boldsymbol{\nu}_t \\ \mathbf{0} \end{bmatrix} \tag{5}$$

where $\boldsymbol{\nu}_t$ is a vector-valued random sample from an independent and identically distributed (i.i.d.) stochastic sequence that accounts for small instantaneous displacements of the head. The second situation can be described by a constant-velocity model

$$\boldsymbol{x}_t^{(CV)} = \begin{bmatrix} \mathbf{1} & \mathbf{1} \\ \mathbf{0} & \mathbf{1} \end{bmatrix} \boldsymbol{x}_{t-1} + \begin{bmatrix} \mathbf{0} \\ \boldsymbol{\eta}_t \end{bmatrix} \tag{6}$$

where $\boldsymbol{\eta}_t$ is a vector-valued sample from another i.i.d. stochastic sequence that accounts for any change in velocity of the head. At a practical level, we do not need to estimate whether the head is in a zero-velocity or constant-velocity mode, since we are only interested in the position and orientation of the head. Instead, these two models simultaneously constitute a mixed prior probability for the motion of the head.

To accommodate both of these motion models, we define the augmented state

$$\boldsymbol{y}_t = \{\boldsymbol{x}_t, \xi_t\} \tag{7}$$

where ξ_t is a binary variable $\{\xi_t : \xi_t \in 0, 1\}$ that specifies the head motion model at time t as

$$\boldsymbol{x}_t = (1 - \xi_t)\boldsymbol{x}_t^{(ZV)} + \xi_t \boldsymbol{x}_t^{(CV)}. \tag{8}$$

We can model ξ_t as a Markov chain, drawing each new sample from a probability distribution $f(\cdot)$ that only depends on the previous state

$$\xi_t \sim f(\xi_t \,|\, \xi_{t-1}). \tag{9}$$

Given this Markov property and the construction of (8), $\boldsymbol{y}_t$ is also a Markov process

$$p(\boldsymbol{y}_t|\boldsymbol{y}_0, \ldots, \boldsymbol{y}_{t-1}) = p(\boldsymbol{y}_t \,|\, \boldsymbol{y}_{t-1}). \tag{10}$$

In a classical tracking problem, the object's state $\boldsymbol{y}_t$ is observed at every time step but assumed to be noisy; hence, the optimal track can be found by maximizing the posterior probability of the movement given the previous states and observations. For a Markovian system that is perturbed by non-Gaussian noise, a sampling importance resampling (SIR) particle filter offers a practical approach that approximates the optimal track as a weighted sum of samples. These samples are drawn from the state transition density [see (10)], and the weight is set proportional to the posterior density of the observation given the samples. In our vision-based tracking problem, instead of observing a noisy sample of the object's state, we observe an image of the object. The observation noise is negligible, but the difficulty lies in inferring the object's state from the image pixels. The solution to this problem can be estimated using a similar SIR construction. We generate a set of state-space samples and use them to render virtual image samples using the fixed-function pipeline of a GPU. Each virtual image can directly be compared with the observed image, and these comparisons can be used to update the particle weights.

Given the existence of a set of N samples with known states $\{\boldsymbol{y}_t^{(l)} : l \in 0, \ldots, N-1\}$, we can devise the observation vector

$$\boldsymbol{z}_t = \begin{bmatrix} d\left(\boldsymbol{y}_t, \boldsymbol{y}_t^{(0)}\right) \\ \vdots \\ d\left(\boldsymbol{y}_t, \boldsymbol{y}_t^{(N-1)}\right) \end{bmatrix} \tag{11}$$

where $d(\boldsymbol{y}, \boldsymbol{y}')$ is an image-based distance metric. As with a classical SIR application, we are required to maintain and update a set of samples with a known state at every time step. We use these samples to update our observation vector, and with (10) and (11), we note that the observation is conditionally independent of all previous states and observations given the current state

$$p(\boldsymbol{z}_t \,|\, \boldsymbol{y}_0, \ldots, \boldsymbol{y}_t, \boldsymbol{z}_0, \ldots, \boldsymbol{z}_{t-1}) = p(\boldsymbol{z}_t \,|\, \boldsymbol{y}_t). \tag{12}$$

As a potential image comparison metric, normalized cross correlation (NCC) provides an appealing approach for comparing two image patches, having the desirable property that it is invariant to affine changes in pixel intensity in either patch. Given two image patches specified as M-dimensional vectors of intensity $\boldsymbol{\phi}$ and $\boldsymbol{\phi}'$, we can specify an NCC-based distance metric as follows:

$$d_{\mathrm{NCC}}(\boldsymbol{\phi}, \boldsymbol{\phi}') = 1 - \frac{1}{\sqrt{\sigma_\phi^2 \sigma_{\phi'}^2}} \sum_{i=0}^{M-1} (\phi_i - \mu_\phi)(\phi'_i - \mu_{\phi'}) \quad (13)$$

where μ_ϕ is the mean of the intensity, and σ_ϕ^2 is the variance of the intensity. The unit constant and the minus sign are introduced to provide a positive distance measure in the range [0, 2].

When the lighting variation is nonaffine (e.g., specular reflections, shadowing, etc.), NCC poorly performs as a global image metric, since the transformation cannot be modeled by a global dc offset and scaling. If the image patches are small enough, however, then it is likely that they will be *locally* affine. As a consequence, better invariance to globally nonuniform lighting can be gained by using the average of a series of P small image patch NCC comparisons spread out over the object of interest. This is the basis of the mean NCC (MNCC) metric that we use in our tracking system as

$$d(\boldsymbol{y}, \boldsymbol{y}') = \frac{1}{P} \sum_{p=0}^{P-1} d_{\mathrm{NCC}}\left(\boldsymbol{\phi}_p, \boldsymbol{\phi}'_p\right). \quad (14)$$

We can directly relate these comparisons to the conditional observation probability if we can model the distribution such that it only depends on the current sample

$$p\left(\boldsymbol{z}_t \middle| \boldsymbol{y}_t^{(l)}\right) \propto h\left(z_{t,l}, \boldsymbol{y}_t^{(l)}\right) \quad (15)$$

where $z_{t,l}$ is the lth component of $\boldsymbol{z}_t$, and $h(\cdot, \cdot)$ is any valid distribution function.

In our head tracking system, we model the observation probability as a truncated Gaussian envelope windowed by the displacement between the current sample state and the sample with the smallest MNCC distance. Denote this latter sample as

$$\boldsymbol{y}_t^{(*)} = \left\{ \boldsymbol{y}^{(l)} : l = \operatorname*{argmax}_l z_{t,l} \right\} \quad (16)$$

and define the state displacement as

$$s(\boldsymbol{y}, \boldsymbol{y}') = d_P(\boldsymbol{y}, \boldsymbol{y}') + \alpha\, d_A(\boldsymbol{y}, \boldsymbol{y}') \quad (17)$$

where $d_P(\cdot, \cdot)$ is the Euclidean distance between the position of the samples, and $d_A(\cdot, \cdot)$ is the angular displacement computed from the inverse cosine of the inner product of a quaternion representation of each sample's orientation. α is a parameter that scales the relative contribution of each measure. From these definitions, we formally define our distribution model as

$$h_t(z, \boldsymbol{y}) = \begin{cases} 0, & T_z < z \\ 0, & T_s < s\left(\boldsymbol{y}, \boldsymbol{y}_t^{(*)}\right) \\ \exp\left(-\frac{1}{2\sigma^2} z^2\right), & \text{otherwise} \end{cases} \quad (18)$$

where T_z and T_s are scalar thresholds, and σ is the standard deviation of the envelope. From qualitative analysis, we use the following parameters in our head tracking system: $\alpha = 0.01$, $T_z = 0.8$, $T_s = 0.012$, and $\sigma = 0.045$.

B. SIR

A particle filter is a Monte Carlo estimation method based on stochastic sampling [38], [39] that, regardless of the state model, converges to the Bayesian optimal solution as the number of samples increases toward infinity. The concept is to choose an appropriate set of weights and point samples

$$\left\{ \left(c_t^{(l)}, \boldsymbol{y}_t^{(l)} \right) : l \in 0, \ldots, N-1 \right\} \quad (19)$$

such that the *a priori* expectation of the state $\boldsymbol{y}_t$ can be approximated from the weighted average [40]

$$\mathrm{E}\left[\boldsymbol{y}_t \mid \boldsymbol{z}_0, \ldots, \boldsymbol{z}_t\right] \approx \sum_{l=0}^{N-1} c_t^{(l)} \boldsymbol{y}_t^{(l)}. \quad (20)$$

Let $p(\boldsymbol{y}_{0:t} | \boldsymbol{z}_{0:t})$ be the posterior probability distribution for all states up until time t. The samples can be drawn from an arbitrary *importance distribution* $\pi(\boldsymbol{y}_{0:t} | \boldsymbol{z}_{0:t})$, and the approximation is valid as long as the weights $c_t^{(l)}$ are proportional to the ratio between the posterior probability distribution and the importance distribution and $\sum_l c_t^{(l)} = 1$.

If we were to continue updating the sample weights, after only a few frames, most of the particle weights would approach zero. To practically account for this, we use a SIR filter that resamples the particles after every iteration. This is accomplished by drawing a new set of samples $\{\overline{\boldsymbol{y}}_t^{(l)} : l \in 0, \ldots, N-1\}$ from the distribution function

$$\rho\left(\overline{\boldsymbol{y}}_t \middle| c_t^{(0:N-1)}, \boldsymbol{y}_t^{(0:N-1)}\right) = \sum_{l=0}^{N-1} c_t^{(l)} \delta\left(\overline{\boldsymbol{y}}_t^{(l)} - \boldsymbol{y}_t^{(l)}\right) \quad (21)$$

where $\delta(\cdot)$ is the Kronecker delta function. After each resampling, the weight of each new sample is set to $1/N$. Given our probabilistic model and choice of $\pi(\boldsymbol{y}_t^{(l)} | \boldsymbol{y}_{0:t-1}^{(l)}, \boldsymbol{z}_{0:t}) = p(\boldsymbol{y}_t^{(l)} | \boldsymbol{y}_{t-1}^{(l)})$, the weight update equation can be reduced to

$$c_t^{(l)} \propto c_{t-1}^{(l)} p\left(\boldsymbol{z}_t | \boldsymbol{y}_t^{(l)}\right). \quad (22)$$

A full iteration of the SIR filter can be described as follows:

1) Update samples: $\boldsymbol{y}_t^{(l)} \sim p(\boldsymbol{y}_t \mid \overline{\boldsymbol{y}}_{t-1}^{(l)})$.
2) Calculate weights: $c_t^{(l)} = (p(\boldsymbol{z}_t | \boldsymbol{y}_t^{(l)}) / \sum_{l=0}^{N-1} p(\boldsymbol{z}_t | \boldsymbol{y}_t^{(l)}))$.
3) Estimate current state: $\widehat{\boldsymbol{x}}_t = \sum_{l=0}^{N-1} c_t^{(l)} \boldsymbol{x}_t^{(l)}$.
4) Resample: $\overline{\boldsymbol{y}}_t^{(l)} \sim \rho(\overline{\boldsymbol{y}}_t | c_t^{(0:N-1)}, \boldsymbol{y}_t^{(0:N-1)})$.

V. Hybrid System Implementation

Our proposed system uses a hybrid pose estimation scheme, combining our static head pose estimator procedure with a real-time implementation of our 3-D model-based tracking algorithm. The static estimator initializes the tracker from a single image frame and, as the head is tracked, continues to

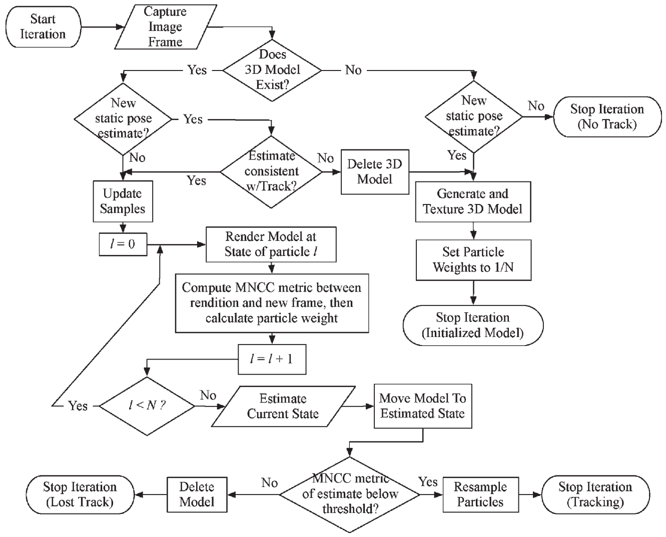

Start Iteration
Capture Image Frame
Does 3D Model Exist?
Yes
No
New static pose estimate?
Yes
No
New static pose estimate?
No
Stop Iteration (No Track)
Yes
Estimate consistent w/Track?
No
Delete 3D Model
Yes
Update Samples
Generate and Texture 3D Model
$I = 0$
Render Model at State of particle I
Set Particle Weights to 1/N
Compute MNCC metric between rendition and new frame, then calculate particle weight
Stop Iteration (Initialized Model)
$I = I + 1$
Yes
$I < N$?
No
Estimate Current State
Move Model To Estimated State
Stop Iteration (Lost Track)
Delete Model
No
MNCC metric of estimate below threshold?
Yes
Resample Particles
Stop Iteration (Tracking)

Fig. 4. Flowchart illustrating one iteration of our head tracking procedure. There are four potential results of each iteration denoted with the phase “stop iteration.”

run in parallel, providing a periodic consistency check. If the tracking confidence falls below a threshold or the consistency check fails, then the static estimator automatically reinitializes the tracker. An overview of the full hybrid system is illustrated in Fig. 4.

A. Camera Perspective

The tracking system has been optimized to run on a GPU. First, we use the intrinsic parameters from our camera to model the perspective projection in the augmented reality. To correctly model the perspective projection of our camera, we must mimic the intrinsic camera parameters in our virtual environment. A camera can be modeled as an ideal perspective camera subject to spherical lens distortion. We refer the reader to [41] for more information. Many software packages are available to estimate and remove the distortion as well as estimate the intrinsic camera parameters that specify a linear projection from world coordinates into camera coordinates. We have calibrated our cameras using checkerboard calibration patterns and the facilities available in the OpenCV software library [42]. Each camera frame is undistorted before any further processing.

To project a 3-D model into the image plane with the same perspective distortion of the camera, we modify the set of projection matrices that define how a point in the virtual world is projected onto a rectilinear image, known as the *viewport*. In OpenGL, this is accomplished using two matrices:

1) ModelView matrix—A linear transformation that accounts for the position and direction of the camera relative to the model.
2) Projection matrix—A linear transformation that projects points into the viewport as *clip coordinates*. The parameters of this matrix affect the projection similar to how a lens affects a camera.

The conversion from an intrinsic matrix to the ModelView and Projection matrices requires a conversion from world coordinates to the *normalized view volume* coordinates used by OpenGL. We refer the reader to [43] for details on this conversion.

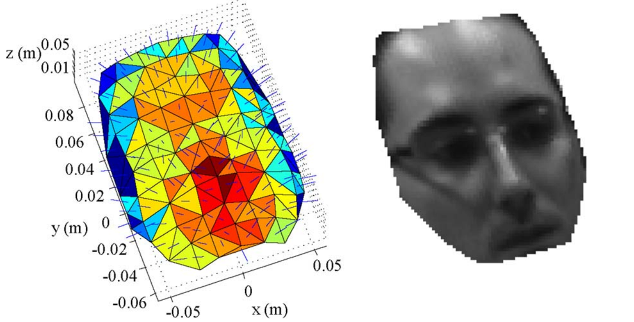


Fig. 5. (Left) Rigid facial model using for initialization and tracking. (Right) Example of the model as rendered by the tracking system.

B. 3-D Model

We represent the driver’s head in our augmented reality framework as a texture-mapped 3-D model. The model consists of 3-D vertices that define a set of convex polygons approximating the surface of the head. Each vertex is assigned a texture coordinate that corresponds to a position in a 2-D image texture.

To create a new model and place it in the environment, we require a new set of polygons and an image texture. For our approach, we use a rigid anthropometric head model shown in Fig. 5. This model was created from a person excluded from the driving experiments, and although this single model is only an approximation of the facial shape of each driver, the texture-based tracking approach does not require a highly accurate fit. We show this in Section VI by comparing the rigid model to individualized models obtained by correspondence-based stereo vision. Once the model is textured, it is placed in the virtual environment with an inverse projection that puts it at the depth that corresponds to the observed width of the detected face. The static pose estimate is used to assign the initial pose angles. To ensure a symmetric view of the head, we only initialize the model if the estimated head pose is within 25° of the center; otherwise, the initialization is skipped until this constraint is satisfied.

C. Sample Generation and GPU-Based MNCC

After each new video frame is captured, it is copied into a texture object on the GPU. For each sample in the particle filter, we generate a virtual representation of the model and calculate the sample weight. To begin, the 3-D head model vertices are rotated and translated as described by the sample state. Next, the model is rendered to an off-screen framebuffer object using the fixed-function GPU pipeline (i.e., the basic procedure for rendering an object with the graphics API). Computing the MNCC distance metric described in (14) requires many computationally intensive pixel calculations. To obtain real-time speeds, we perform the calculation with the programmable pipeline of the GPU using the OpenGL Shading Language [44]. We render the vertices as individual points that compute the NCC of a local image patch using the programmable pipeline. More specifically, we compute the NCC with a *vertex shader*,

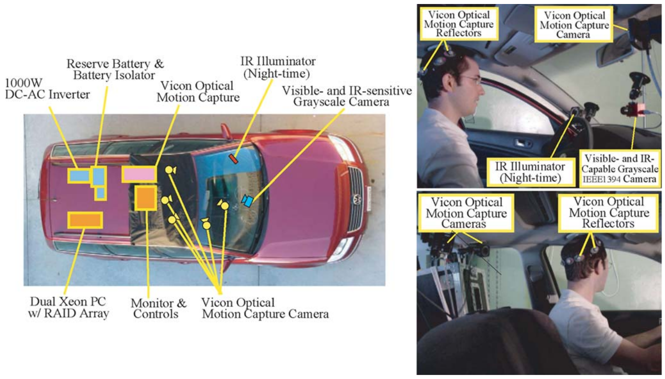


Fig. 6. LISA-P experimental testbed is a modified Volkswagen Passat equipped with a mobile computing platform and sensors for motion and video capture [45].

which is a program that operates on each vertex as it passes through the GPU pipeline. This enables hundreds of NCC windows to be computed in parallel. A full description of this GPU optimization is available in [43].

VI. Data and Evaluation: Laboratory for Intelligent and Safe Automobiles-P Testbed

The Laboratory for Intelligent and Safe Automobiles (LISA)-P experimental testbed, as shown in Fig. 6, is used to collect real-world test data. The vehicle is a modified Volkswagen Passat. An IEEE1394 camera mounted on the windshield is used to capture face data, as shown in Fig. 6. This camera captures a 640 × 480 pixel grayscale video stream at 30 fps, and like most CCD imagers, it is naturally sensitive to both visible and near-IR light. For our specific camera, we were required to physically remove an IR filter installed above the imager.

For the purpose of illuminating the driver's face and stabilizing the lighting conditions at nighttime, a near-IR illuminator (light-emitting-diode array with plastic diffuser) was placed on the leftmost part of the windshield. Since the emitted light is not part of the visible spectrum, it does not serve as a distraction or cause any glare for the driver. In addition, the vehicle is instrumented with a Vicon optical motion capture system, with five sensors placed in various locations around the driver's head. This marker-based system is used to gather precise ground truth head pose data for evaluation. To prevent the reflective markers from appearing in the video, we created an unobtrusive headpiece for the subjects to wear on the back of their head, as shown in Fig. 6.

For the automotive experiments, we asked 14 subjects to drive the LISA-P while wearing the motion capture headpiece. The subjects consisted of 11 males and three females, spanning Caucasian, Asian, and south-Asian descent. The subjects ranged from 15 to 53 years of age, and five of them wore glasses.

Each of the subjects drove the vehicle on different round-trip routes through the University of California campus at different times, including drives from the morning, afternoon, dusk, and night. The cameras were set to autogain and autoexposure, but these adjustments have to compete with ever-shifting lighting conditions, and dramatic lighting shifts (e.g., sunlight diffracting around the driver or headlights from a neighboring vehicle) on occasion would completely saturate the image. All of these situations remain part of our experimental data, as they are typical phenomena that occur in natural driving.

TABLE I
Comparison of Mean Absolute Error Between Head Pose Estimation Approaches

	Laboratory		Day Driving		Night Driving	
Method	*Pitch*	*Yaw*	*Pitch*	*Yaw*	*Pitch*	*Yaw*
Norm. Cross-corr.	**5.19°**	21.25°	11.30°	14.90°	5.94°	16.49°
Grad. PCA + SVR	5.72°	13.46°	**3.62°**	12.19°	5.19°	13.11°
LGO Hist. + SVR	5.58°	**6.40°**	3.99°	**9.28°**	**5.18°**	**7.74°**

The automobile was set up to collect data during two periods: 1) half during the summer and 2) half during the winter. The placement of the cameras mildly varies between these two setups. The drives averaged 8 min in duration, amounting to approximately 200 000 video frames in all.

Experiment 1—Static Head Pose Comparison: We compare our static head pose estimation procedure to two alternative approaches for estimating the pitch and yaw of the driver's head. The first is a prototype matching scheme that uses NCC to compare the driver's face to each of the views in our training data. To make the system more robust to noise, we take the mean of the cross-correlation score for all the training images that share the same discrete pose, and we estimate the head pose as the pitch and yaw corresponding to the maximum score after bicubic interpolation.

The second comparative head pose estimation system is our implementation of the gradient PCA system described by Li *et al.* [18]. We chose this work for comparison since it is the most similar to our proposed system and it is capable of high accuracy and speed. This approach also uses two SVRs to estimate pitch and yaw. Instead of LGO histograms, the input to each regressor is the raw horizontal and vertical image gradient reduced to a 50-D vector using principal component analysis. The PCA basis is derived from the training data.

For both of these comparative approaches, we use the same array of Adaboost cascades described in Section III-A to locate and normalize the region of the image corresponding to the driver's face.

The data used for this evaluation is a 1-min excerpt from each of the six drives: two during the daytime and four during the nighttime. In addition, we provide a comparison for an indoor scenario to evaluate whether the differences between the algorithms are specific to the automotive imagery. For indoor experiments, ten people were asked to sit on a chair against a white background while facing an IEEE1394 video camera. Behind the camera, a projector displayed a grid of points on a screen in front of the subject, each point representing a specific pose at 5° intervals spanning $(-30^\circ, 20^\circ)$ in pitch and $(-80^\circ, 80^\circ)$ in yaw. Within this grid, we displayed an active cursor, which corresponds to the subject's current head pose as measured by the motion capture system. When a subject moved his/her head to any of the 363 grid point locations for the first time, the point would change color, and the camera would capture an image of the subject. In this fashion, we obtained a uniform sampling of all ten subjects across the pose space.

The results of these experiments are found in Table I. Here, we quantify each approach by the mean absolute error in pitch and yaw between the motion capture reference and the estimated orientation. In the laboratory experiment, all three systems provide a comparable level of error in pitch, and the LGO histograms demonstrate a 7.06° reduction in yaw error over the gradient PCA approach and a 14.85° reduction over the NCC approach. In the driving experiments, our algorithm again outperforms the other approaches in absolute yaw error: 9.28° compared with 14.90° and 12.19° during the daytime experiment, and 7.74° compared with 16.49° and 13.11° during the nighttime drives. We attribute the general improvement in yaw estimation with LGO histograms to the explicit invariance they provide to positional and orientational error caused by automatic face detection and localization. Although all three approaches are invariant to affine lighting changes, the normalized cross-correlation approach shows a significant reduction in pitch estimation during the daytime drives. We attribute this to the inability of this template matching approach to operate with nonaffine lighting caused by sunlight. The SVR-based approaches in comparison do not show a decrease in accuracy from indoors to outdoors. We attribute this to the representation ability of the regressors, which learn models that account for this lighting variation. Examples of all three systems along with the ground truth data are presented in Fig. 3, and although the day driving experiment had better pitch accuracy than the laboratory experiment, we attribute this in part to the Gaussian-like distribution of head orientations in the driving experiments compared with the uniform pitch variation in the laboratory evaluation. As shown in Fig. 8, the pitch error is typically smaller for the near-frontal orientations that are frequent during driving.

Experiment 2—Anthropometric 3-D Model Evaluation: To meet our design requirement for a monocular approach, we generate a textured 3-D model of the head from a 2-D image. This is accomplished by placing a generic anthropometric facial mesh in the projected location of the detected face at a depth that corresponds to the perspective width of the face. The texture is assigned to the model by projecting the first image of the tracking sequence onto this mesh. To verify that this generic model is a sufficient approximation for tracking, we compare it to individualized texture models that are generated using a commercial stereo correspondence algorithm to estimate the 3-D shape of the driver's face [46].

For this comparison, we evaluate the tracking system on 1-min excerpts from six of the drives in which we also captured a second video stream that can provide a binocular view of the driver. This allows us to create a stereo depth map of the driver's face. By sampling the map at a 10 × 10 pixel interval and computing the Delaunay triangulation, we create a triangular mesh that corresponds to the surface of the face.[1] A global center and orientation is assigned in the same fashion as was done for the generic model.

TABLE II
COMPARISON OF MONOCULAR GENERIC 3-D MODEL TO STEREO-BASED INDIVIDUALIZED MODELS USING MEAN ABSOLUTE ERROR

	Translation (cm)			Rotation (deg.)		
Model	*x*	*y*	*z*	*Pitch*	*Yaw*	*Roll*
Generic Model	0.88	1.35	1.78	4.10	5.64	2.42
Individualized Models	0.87	1.54	1.95	4.08	5.78	2.91

TABLE III
ISOLATED ERROR FOR STATIC HEAD POSE ESTIMATION

	Rotation (degrees)		
Statistic	*Pitch*	*Yaw*	*Roll*
Mean absolute error	5.71	7.53	7.34
Std. deviation of error	7.93	10.93	10.39

TABLE IV
ISOLATED ERROR OF TRACKING ALGORITHM

	Translation (cm)			Rotation (degrees)		
Statistic	*x*	*y*	*z*	*Pitch*	*Yaw*	*Roll*
Mean absolute error	0.92	0.88	1.44	3.39	4.67	2.38
Std. deviation of error	1.87	1.22	2.37	4.71	6.89	3.74

In our comparison, we ignore the error generated by lost tracks and compute the mean absolute error for all of the successfully tracked frames, which we define as any track where the estimate is within 30.0° of the true pitch, yaw, and roll. The results of this comparison are presented in Table II. This shows comparable tracking accuracy with either model, with the generic model slightly outperforming the individualized models slightly in yaw and roll estimation. As we would expect individualized methods to perform better than the generic model, we attribute this contradictory result to occasional correspondence errors in the stereo model that are potentially more detrimental to the tracker than the use of a single generic facial shape. From an implementation perspective, both fixed and dynamic models yield comparable performance, but the fixed-model approach has the advantage of using a single camera.

Experiment 3—Hybrid System Evaluation: In this experiment, we evaluate our tracking system on the full video footage obtained from all 14 drivers. To train the static head pose estimator, we separately extracted a uniform sampling of the pose space for pitch, yaw, and roll (approximately 300 images from each 10-degree interval where available, and all of the data from the intervals with fewer than 300 images) and used a cross-validation scheme to train with the data from 13 of the subjects while leaving the remainder for evaluation. This was repeated for every all-but-one combination.

We first present the results for the head pose estimator and the tracking algorithm independent of each other. Table III shows the mean absolute error and the standard deviation of the error for the static head pose estimator, and Table IV provides a similar treatment for the tracking system. To compute these latter statistics, the mean position and orientation are subtracted from the ground truth and the estimated track before calculating the mean absolute error between the two. We also exclude frames in which the tracker has suffered catastrophic error due to a lost track, which we again quantify as the frames where the tracked head orientation deviates more than 30° from the true pitch, yaw, or roll. Since this is the first system to be evaluated on these data, we cannot directly compare these results to other systems. Nevertheless, our tracking results on these challenging data are within one or two degrees of the error from prior systems evaluated on much simpler data sets (i.e.,

[1] Any points in the mesh that lie 20 cm beyond the median depth are considered as outliers and are discarded.

TABLE V
COMBINED ANGULAR ERROR FROM INITIALIZATION AND TRACKING

	Rotation (degrees)		
Statistic	*Pitch*	*Yaw*	*Roll*
Mean absolute error	8.57	11.24	8.29
Standard deviation of error	16.43	16.90	15.27

TABLE VI
COMPARISON OF STATIONARY JITTER BETWEEN STATIC POSE ESTIMATOR AND HYBRID SYSTEM

	Rotation (degrees)		
Method	*Pitch*	*Yaw*	*Roll*
Static Pose Estimation Only	5.75	8.20	6.72
Hybrid Pose Estimation	1.64	2.08	1.55

indoors with only a few subjects) [14], [21]. It is worth noting that these prior systems also require calibrated stereo cameras for depth information, whereas our system uses a single camera. From these tables, one can observe that the pose errors for the tracking system are substantially smaller than the pose errors for static head pose estimation procedure.

To evaluate the combined error for the full hybrid system, we directly compare the output of the system to the motion capture ground truth. These results are presented in Table V. This table only contains angular evaluation, since the ground truth is ambiguous as to the exact position of the face, which is not the same as the position of the motion capture headpiece. This result combines the errors of the static head pose estimator and the tracker, and it also contains the errors from any lost tracks. Although the mean absolute error is larger than the static head pose estimator by itself (as should be expected), the quality of the track is much better since the actual motion of the head is better captured by the addition of the visual tracking algorithm. We can quantify this in terms of the observed jitter. We define jitter as the mean absolute change in orientation between two successive frames (i.e., the derivative of the estimate) while the head is stationary. The ground truth is used to discover the stationary frames, and the jitter is presented in Table VI. The hybrid approach shows a large reduction in jitter, as it is very good at providing a smooth and accurate track of the head motion. This is important for applications in driver intent, where the motion of the driver's head provides cues to his/her intentions (e.g., predicting when the driver is about to perform a lane change). In Fig. 7, we plot the stationary jitter as a function of the pose angle. These plots show that the static pose estimator exhibits more jitter for poses that are far off-center, whereas the hybrid system is fairly consistent across the pose space.

To show the influence of head angle on our system, we plotted the pose estimate and the standard deviation for successful tracks as a function of the true orientations in Fig. 8. The histograms on the top of each plot show the relative frequency of each pose angle. For most of the pose space, the estimate is within one standard deviation of the true pose. The static head pose estimator exhibits a regular bias toward zero that causes the curve to deviate from a pure linear slope, but the error variance is relatively stable across the pose space.

To provide an example of lost tracks and reinitialization, we include a cross-sectional plot of head yaw for a challenging 1-min video excerpt in Fig. 10. Here, the true yaw is shown alongside the estimated yaw after removing any bias from the static pose estimator. In this excerpt, there are two situations in which the track is lost and reinitialized. Beginning from a successful track, the system closely follows the head until approximately 23 s. At this point, the driver makes an abrupt turn to the left and then an abrupt turn to the right. The track is lost at this point and then regained by reinitialization at about 28 s. A similar process occurs at 44 s. In these cases, the track is lost at the yaw extremes. This is difficult for the tracker since these far rotations project less of the model than frontal views. In addition, at 48 s, the tracker fails to keep up with the fast head movement but still maintains the track when the movement slows down.

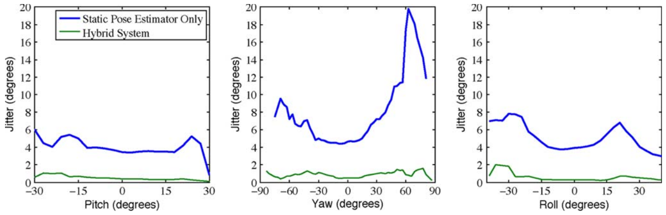


Fig. 7. Stationary jitter as a function of head position.

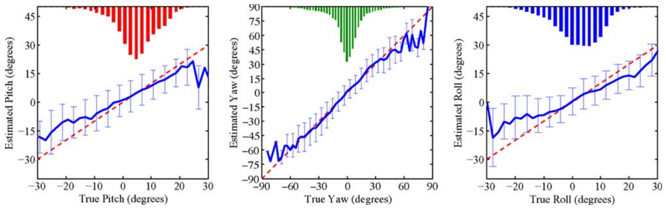


Fig. 8. Mean estimated head position as a function of the true angle. The error bars indicate one standard deviation, and the histograms show the frequency of each angle.

Images of a tracking sequence are provided in Fig. 9. We also include example videos of the running system as supplementary material. We encourage the readers to view these videos as they provide better visualization of the system than is possible with the images alone.

VII. CONCLUSION

Robust systems for observing driver behavior will play a key role in the development of advanced driver assistance systems. In combination with environmental sensors, cars can be designed with the ability to supplement driver's awareness, preempting and preventing hazardous situations. In this paper, we have presented new algorithms for automotive head pose estimation and tracking, since head pose is a strong indicator of a driver's field of view and current focus of attention. The system satisfies all of our design criteria as it only requires monocular video for autonomous, real-time, identity-invariant, and lighting-invariant driver head pose estimation. It is an advancement in the state of the art, providing fine head pose estimation and ease of use.

We contribute two new processes that represent advances over previous head pose estimation approaches. Using LGO histograms to tolerate deviations caused by scale, position, rotation, and lighting, we demonstrated that they provide superior input to SVRs for robust head pose estimation in two degrees

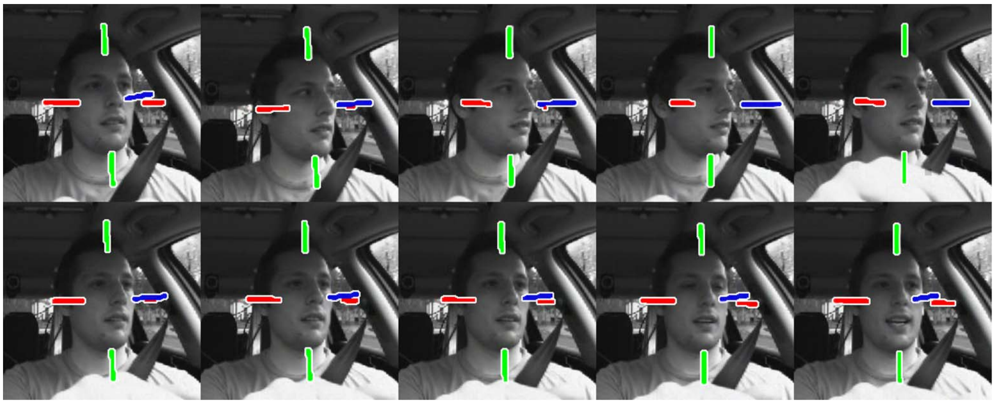


Fig. 9. Example images from a daytime tracking sequence. The images have been cropped around the driver's face to highlight the tracking estimate, which is indicated by the overlaid 3-D axes.

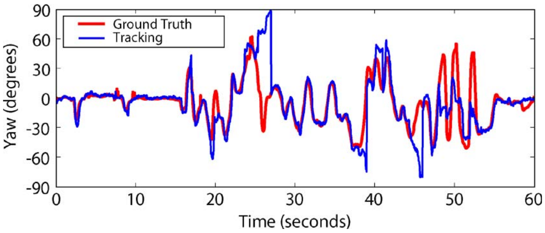


Fig. 10. Cross section of yaw during a daytime tracking sequence. The tracking bias is removed to exclude initialization error.

of freedom. The output of this static head pose estimator is used to reinitialize our particle filter-based head tracker. This real-time tracker updates a 3-D model of the head using a set of appearance-based comparisons that estimate the movement that minimizes the difference between a virtual projection of the model and the subsequent image frame.

Further extensions to this system could focus on model augmentation, since the initial model represents only a slice of the head that was visible from the perspective of a single camera when the model was created. As the head rotates, this region shifts out of view until there is very little texture remaining to continue the tracking. This effect is visible in Fig. 10, which shows how the track can become less reliable as the yaw of the head approaches 90° in either direction. As a possible solution, the model can be augmented by adding additional sets of polygons and textures. During tracking, if the rotation angle between the sample and the initial model exceeds a threshold and the MNCC score is sufficiently large to indicate an accurate track, then the initialization step can be repeated to add new polygons with a new texture to augment the original model. Care should be taken to prevent adding polygons that can overlap the existing model, and during this augmentation process, the global position and orientation do not need to be reestimated, since the are already established by the particle filter.

In conclusion, the system consists of a new method for estimating the pose of a human head that overcomes the difficulties inherent with varying lighting conditions in a moving car.

Acknowledgment

The authors would like to thank the Volkswagen Electronics Research Laboratory and the U.C. Discovery program for their support, A. Doshi for his help with experimental setup and data collection, S. Cheng for his contributions to the design and development of the LISA-P testbed and motion capture system, and the rest of their team members at the Computer Vision and Robotics Research Laboratory for their participation as test subjects and for their continuing support, comments, and assistance.

References

[1] T. Rueda-Domingo, P. Lardelli-Claret, J. L. del Castillo, J. Jiménez-Moleón, M. Garciá-Martín, and A. Bueno-Cavanillas, "The influence of passengers on the risk of the driver causing a car collision in Spain," *Accident Anal. Prevention*, vol. 36, no. 3, pp. 481–489, 2004.

[2] A. Doshi and M. M. Trivedi, "A novel active heads-up display for driver assistance," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 39, no. 1, pp. 85–93, Feb. 2009.

[3] M. Trivedi, T. Gandhi, and J. McCall, "Looking-in and looking-out of a vehicle: Computer-vision-based enhanced vehicle safety," *IEEE Trans. Intell. Transp. Syst.*, vol. 8, no. 1, pp. 108–120, Mar. 2007.

[4] S. Cheng and M. Trivedi, "Holistic sensing and dynamic active displays," *Computer*, vol. 40, no. 5, pp. 60–68, May 2007.

[5] R. Hammoud, A. Wilhelm, P. Malawey, and G. Witt, "Efficient real-time algorithms for eye state and head pose tracking in Advanced Driver Support systems," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2005, vol. 2, pp. 20–25.

[6] R. Hammoud, *Passive Eye Monitoring: Algorithms, Applications and Experiments*, 1st ed. Berlin, Germany: Springer-Verlag, 2008, ser. Signals and Communication Technology.

[7] A. Doshi and M. M. Trivedi, "On the roles of eye gaze and head pose in predicting driver's intent to change lanes," *IEEE Trans. Intell. Transp. Syst.*, vol. 10, no. 3, pp. 453–462, Sep. 2009.

[8] S. Langton, H. Honeyman, and E. Tessler, "The influence of head contour and nose angle on the perception of eye-gaze direction," *Percept. Psychophys.*, vol. 66, no. 5, pp. 752–771, Jul. 2004.

[9] L. Bergasa, J. Nuevo, M. Sotelo, R. Barea, and M. Lopez, "Real-time system for monitoring driver vigilance," *IEEE Trans. Intell. Transp. Syst.*, vol. 7, no. 1, pp. 63–77, Mar. 2006.

[10] P. Smith, M. Shah, and N. da Vitoria Lobo, "Determining driver visual attention with one camera," *IEEE Trans. Intell. Transp. Syst.*, vol. 4, no. 4, pp. 205–218, Dec. 2003.

[11] J. McCall, D. Wipf, M. M. Trivedi, and B. Rao, "Lane change intent analysis using robust operators and sparse Bayesian learning," *IEEE Trans. Intell. Transp. Syst.*, vol. 8, no. 3, pp. 431–440, Sep. 2007.

[12] E. Murphy-Chutorian and M. Trivedi, "Head pose estimation in computer vision: A survey," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, no. 4, pp. 607–626, Apr. 2009.

[13] T. Jebara and A. Pentland, "Parameterized structure from motion for 3d adaptive feedback tracking of faces," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 1997, pp. 144–150.

[14] L.-P. Morency, A. Rahimi, and T. Darrell, "Adaptive view-based appearance models," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2003, pp. 803–810.

[15] K. Huang and M. Trivedi, "Robust real-time detection, tracking, and pose estimation of faces in video streams," in *Proc. Int. Conf. Pattern Recog.*, 2004, pp. 965–968.

[16] R. Rae and H. Ritter, "Recognition of human head orientation based on artificial neural networks," *IEEE Trans. Neural Netw.*, vol. 9, no. 2, pp. 257–265, Mar. 1998.

[17] R. Stiefelhagen, J. Yang, and A. Waibel, "Modeling focus of attention for meeting indexing based on multiple cues," *IEEE Trans. Neural Netw.*, vol. 13, no. 4, pp. 928–938, Jul. 2002.

[18] Y. Li, S. Gong, and H. Liddell, "Support vector regression and classification based multi-view face detection and recognition," in *Proc. IEEE Int. Conf. Autom. Face Gesture Recog.*, 2000, pp. 300–305.

[19] F. Dornaika and F. Davoine, "Head and facial animation tracking using appearance-adaptive models and particle filters," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2004, pp. 153–162.

[20] J. Tu, T. Huang, and H. Tao, "Accurate head pose tracking in low resolution video," in *Proc. IEEE Int. Conf. Autom. Face Gesture Recog.*, 2006, pp. 573–578.

[21] K. Oka, Y. Sato, Y. Nakanishi, and H. Koike, "Head pose estimation system based on particle filtering with adaptive diffusion control," in *Proc. IAPR Conf. Mach. Vis. Appl.*, 2005, pp. 586–589.

[22] K. Oka and Y. Sato, "Real-time modeling of face deformation for 3d head pose estimation," in *Proc. IEEE Int. Workshop Anal. Model. Faces Gestures*, 2005, pp. 308–320.

[23] E. Murphy-Chutorian and M. M. Trivedi, "3d tracking and dynamic analysis of human head movements and attentional targets," in *Proc. ACM/IEEE Int. Conf. Distrib. Smart Cameras*, 2008, pp. 1–8.

[24] R. Pappu and P. Beardsley, "A qualitative approach to classifying gaze direction," in *Proc. IEEE Int. Conf. Autom. Face Gesture Recog.*, 1998, pp. 160–165.

[25] Y. Zhu and K. Fujimura, "Head pose estimation for driver monitoring," in *Proc. IEEE Intell. Veh. Symp.*, 2004, pp. 501–506.

[26] J. Wu and M. Trivedi, "Visual modules for head gesture analysis in intelligent vehicle systems," in *Proc. IEEE Intell. Veh. Symp.*, 2006, pp. 13–18.

[27] K. Huang and M. Trivedi, "Driver head pose and view estimation with single omnidirectional video stream," *Autom. Remote Control*, vol. 25, pp. 821–837, 2006.

[28] Z. Guo, H. Liu, Q. Wang, and J. Yang, "A fast algorithm face detection and head pose estimation for driver assistant system," in *Proc. Int. Conf. Signal Process.*, 2006, vol. 3.

[29] S. Baker, I. Matthews, J. Xiao, R. Gross, T. Kanade, and T. Ishikawa, "Real-time non-rigid driver head tracking for driver mental state estimation," in *Proc. 11th World Congr. Intell. Transp. Syst.*, 2004.

[30] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2001, pp. 511–518.

[31] R. Lienhart and J. Maydt, "An extended set of Haar-like features for rapid object detection," in *Proc. IEEE Int. Conf. Image Process.*, 2002, vol. 1, pp. 900–903.

[32] D. Lowe, "Distinctive image features from scale-invariant keypoints," *Int. J. Comput. Vis.*, vol. 60, no. 2, pp. 91–110, 2004.

[33] K. Mikolajczyk and C. Schmid, "A performance evaluation of local descriptors," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 27, no. 10, pp. 1615–1630, Oct. 2005.

[34] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2005, pp. 886–893.

[35] H. Drucker, C. Burges, L. Kaufman, A. Smola, and V. Vapnik, "Support vector regression machines," in *Proc. Adv. Neural Inf. Process. Syst.*, 1996, pp. 155–161.

[36] A. Smola and B. Scholkopf, "A tutorial on support vector regression," *Stat. Comput.*, vol. 14, no. 3, pp. 199–222, Aug. 2004.

[37] C.-C. Chang and C.-J. Lin, LIBSVM: A library for support vector machines, 2001. [Online]. Available: http://www.csie.ntu.edu.tw/~cjlin/libsvm

[38] S. Haykin, *Adaptive Filter Theory*, 4th ed. Englewood Cliffs, NJ: Prentice-Hall, 2002.

[39] S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp, "A tutorial on particle filters for on-line non-linear/non-Gaussian Bayesian tracking," *IEEE Trans. Signal Process.*, vol. 50, no. 2, pp. 174–188, Feb. 2002.

[40] N. G. Arnaud Doucet and N. de Freitas, *Sequential Monte Carlo Methods in Practice*. New York: Springer-Verlag, 2001.

[41] R. I. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, 2nd ed. Cambridge, MA: Cambridge Univ. Press, 2004.

[42] OpenCV Computer Vision Library, 2007. [Online]. Available: http://sourceforge.net/projects/opencvlibrary/

[43] E. Murphy-Chutorian and M. M. Trivedi, "Particle filtering with rendered models: A two pass approach to multi-object 3D tracking with the GPU," in *Proc. Comput. Vis. Pattern Recog. Workshop, Vis. Comput. Vis. GPUs*, 2008, pp. 1–8.

[44] J. Kessenich, D. Baldwin, and R. Rost, The OpenGL Shading Language, ver. language 1.2, 3DLabs, Inc. Ltd., Milpitas, CA, 2006.

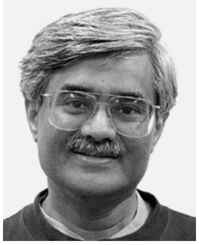
[45] S. Cheng and M. Trivedi, "Turn-intent analysis using body pose for intelligent driver assistance," *Pervasive Comput.*, vol. 5, no. 4, pp. 28–37, Oct. 2006.

[46] V. Design, Small Vision System Software. [Online]. Available: http://www.ai.sri.com/~konolige/svs/svs.htm

Erik Murphy-Chutorian (M'09) received the B.A. degree in engineering physics from Dartmouth College, Hanover, NH, in 2002 and the M.S. and Ph.D. degrees in electrical and computer engineering from the University of California, San Diego (UCSD), La Jolla, in 2005 and 2009, respectively.

At UCSD, he tackled problems in computer vision, including object recognition, invariant region detection, visual tracking, and head pose estimation. He has designed and implemented numerous real-time systems for human–computer interaction, intelligent environments, and driver assistance. He is currently a Software Engineer with Google Inc., Mountain View, CA, where he works on image content processing.

Dr. Murphy-Chutorian is actively involved as a Reviewer and program committee member for numerous computer vision and intelligent transportation publications and serves on the Computer Vision Technical Expert Task Group for the National Academies Transportation Research Board's Strategic Highway Research Program.



Mohan Manubhai Trivedi (F'09) received the B.E. degree (with honors) from the Birla Institute of Technology and Science, Pilani, India, and the Ph.D. degree from Utah State University, Logan.

He is currently a Professor of electrical and computer engineering and the Founding Director of the Computer Vision and Robotics Research Laboratory, University of California, San Diego (UCSD), La Jolla. He has established the Laboratory for Intelligent and Safe Automobiles, UCSD, to pursue a multidisciplinary research agenda. He and his team are currently pursuing research on machine and human perception, active machine learning, distributed video systems, multimodal affect and gesture analysis, human-centered interfaces, intelligent driver assistance, and transportation systems. He regularly serves as a consultant to industry and government agencies in the U.S. and abroad. He has given over 50 keynote/plenary talks. He served as the Editor-in-Chief of the *Machine Vision and Applications Journal* and is currently an Editor for *Image and Vision Computing*.

Prof. Trivedi is also currently an Editor for the IEEE TRANSACTIONS ON INTELLIGENT TRANSPORTATION SYSTEMS. He serves as the General Chair for the 2010 IEEE Intelligent Vehicles Symposium IV. He has received the Distinguished Alumnus Award from Utah State University, the Pioneer Award and Meritorious Service Award from the IEEE Computer Society, and a number of "Best Paper" Awards. He is a Fellow of The International Society for Optical Engineers.