Interactive Model-Based Vehicle Tracking

by

Warren F. Gardner, Daryl T. Lawton

GIT-GVU-95-33
September 1995

Graphics, Visualization & Usability Center

Georgia Institute of Technology
Atlanta GA 30332-0280
Interactive Model-Based Vehicle Tracking

Warren F. Gardner       Daryl T. Lawton

Abstract

This paper describes an interactive model-based vision system for vehicle tracking. A human specifies a limited amount of information in the form of object models, which establish a context for autonomous interpretation of scenes containing moving vehicles. The system is able to successfully track vehicles under complex outdoor conditions through the use of gravity, vehicle, and road models. Results are presented from several image sequences shot with hand-held uncalibrated cameras.

1 Introduction

The ability to track vehicles in outdoor scenes is important for a variety of applications, yet this remains a difficult problem. Image understanding systems that are able to operate in complex, natural domains have been largely unsuccessful to date. There remain unresolved and fundamental difficulties in terms of the necessary computational power, the required complexity of perceptual systems which can operate in outdoor environments, and the corresponding complexity of planning and reasoning systems. Previous work has addressed many of these problems by stressing the importance of telerobotic and interactive systems [13, 16]. This is a realistic approach to fielding advanced technology in the short term, and also provides a long term framework for developing autonomous systems. An interactive, semi-autonomous system can significantly amplify the capabilities of a human, and also yields an evolutionary approach as autonomous system capabilities are developed and begin to replace human controlled functions. This interactive approach to vision is discussed more fully in [10].
The vehicle tracking system presented in this paper makes use of gravity and vehicle models, along with a vehicle motion model to constrain subsequent processing. Using models to constrain possible solutions to the motion problem is a technique that has been employed by several researchers. The motion estimation system described in [17] uses a priori object knowledge to solve for the location of the object in 3D from a monocular image sequence. The object model is not discussed, but is apparently a single parameter used to fix the scale factor. The Kalman Filter is used for motion tracking and object localization, but is not used for prediction and feature extraction.

A much more complete model-based tracking system is described by Gennery [5]. In addition to the object geometry, the reflectivity of faces on the object is modeled. The object position and orientation is extrapolated during tracking, and the object model is projected onto the image plane. The predicted position of features is used to constrain feature extraction in subsequent image frames. The filter is essentially a Kalman Filter with a constant linear and angular velocity kinematic model. A model which assumes constant angular momentum rather than constant angular velocity is derived, but not implemented. The techniques are sufficiently general to allow the use of multiple cameras. Results computed from both monocular and stereo image sequences are presented.

Model-based motion estimation in the domain of traffic scenes is addressed in [6, 7]. In [7] vehicle trajectories are automatically characterized in terms of natural language by associating motion verbs with trajectory segments. The vehicle models used in this system are simple rectangles. The tracking takes place in 2D on the image plane, although the trajectories are projected onto a known street plane before associating motion verbs. A much more complete system is presented in [6] where a parameterized vehicle model is tracked in monocular image sequences. The vehicle motion is assumed to be planar, and the street plane is known a priori. Linear and angular velocity is assumed constant,
and motion estimation is performed using the Kalman Filter. Measurements are made by comparing extracted lines to the projection of the vehicle model, and lines extracted from vehicle shadows are also used in the matching process. Vehicle shadow predictions are made using the vehicle model, the known street plane, and a light source direction which is also known a priori.

Another novel vehicle motion estimation technique is presented in [11]. This technique assumes that the motion consists of a rotation about the vehicle center followed by a translation along the main axis of the vehicle. A linear algorithm is presented using this motion model along with an assumption of constant motion. The algorithm is tested using an image sequence for which existing two-view point correspondence and three-view line correspondence algorithms failed to produce reasonable results.

The most impressive demonstration of motion analysis in the vehicle domain has been the work of Dickmanns [2, 3]. A complex geometric road model is combined with a kinematic vehicle ego-motion model to produce a system capable of traveling at speeds of up to 100 km/h. In related work a fairly complex generic vehicle model containing 78 shape parameters was developed [15]. Through symmetries and other geometric constraints, the number of independent parameters was reduced to 12. Image measurements are in the form of straight line segments extracted from image sequences. Like the system described in [6], the vehicle motion is assumed to take place in a known plane, and is estimated using the Kalman Filter. However, assumptions of constant linear and angular acceleration result in a more complex motion model. Currently the system has only been tested on synthetic images. This idea of generic object models is powerful and should be the subject of future investigation.

Almost all the vehicle tracking systems available involve single objects moving under indoor or controlled lighting conditions. The systems which have been tested under complex outdoor conditions are described in [2, 3, 6, 11]. In [11] the algorithm is shown
to be robust for vehicle motion that is approximately constant, and the authors state in
the conclusion that they are currently investigating other special cases of motion. The
system presented in [2, 3] is capable of real-time performance under complex lighting
conditions, but the system works only with single objects. In [6] multiple objects are
tracked in complex outdoor scenes. The object motion is assumed to be planar, and
the plane with respect to the stationary camera must be known a priori. The system
also uses the light source direction, if this information is available. This system has
successfully tracked vehicles that span regions as small as 20 x 40 pixels. It is not clear
from this paper how errors in the a priori parameters would affect the resulting vehicle
trajectories.

2 Object Models

The outdoor vehicle domain contains complex lighting conditions and independently
moving objects, yet there are also constraints which can be exploited through temporal
and geometric models. Three models were developed to enable the tracking of vehicles
in outdoor environments. The objects modeled were roads, gravity, and vehicles.

2.1 Road Model

Knowledge of the position of a road is obviously beneficial to a vehicle tracking system.
Information about road locations can restrict the amount of processing necessary to
detect and track vehicles. The road model consists of a sequence of 2D or 3D line
segments. A road is instantiated by drawing this sequence of lines on top of an image.
This can be done by picking points with a mouse, or by tracking a vehicle moving along
the road. Figure 1 shows a 2D road model that was instantiated by a user with a mouse.
A vehicle tracker which makes use of this road model is presented in [10].
2.2 Gravity Model

Knowledge of the direction of gravity is useful in constraining the orientations of other objects in the world model. The position and orientation of falling objects are obviously affected by gravity. Man-made objects such as buildings lie parallel to the direction of gravity, and vehicles are constrained to travel at angles close to the plane perpendicular to the gravity direction. This last constraint was exploited in this vehicle tracking system.

The gravity direction is represented by a 3D vector. This direction is presented graphically in two ways. The vector is given a planar base and drawn within a sphere to aid the user in determining its current 3D position as shown in Figure 2. A vector in the direction opposite gravity is also drawn in a lighter color to aid the user in determining the current gravity direction when the planar base interferes with the view of the gravity vector. This representation also acts as the user interface for the gravity model. A user positions the gravity vector within the sphere by clicking on the vector and dragging it with a mouse. The 3D gravity vector is also shown by projecting the direction onto an image at different positions, creating the 2D gravity field shown in Figure 3. As a
Figure 2: User interface for the gravity model

Figure 3: Gravity model projected onto an image
user moves the gravity vector within the sphere, the gravity field overlaying the image reflects the change in direction.

2.3 Vehicle Model

The most important model for a vehicle tracking system is certainly the vehicle model. The vehicle model geometry consists of six parameters: length, width, height, hood length, hood height, and wheel height. A perspective view of the vehicle model is shown in Figure 4. A user instantiates a vehicle within the world by placing this graphical model in the image. The six degrees of freedom of the graphical model are manipulated using a mouse.

Inter-object constraints are represented implicitly within each of the object models. The gravity model defines a ground plane which can be used to restrict the motion of the vehicle model. Instantiation of the gravity model places constraints on the orientation of the vehicle model. If a vehicle model is instantiated, gravity will initially be aligned with the vehicle axis. As the gravity direction is changed, the vehicle orientation also changes. When a vehicle model is manipulated in the presence of a gravity model, the user will be restricted to rotations about the gravitational axis.
3 Local Translation-Based Vehicle Tracking

Vehicles have a limited turning radius, resulting in an axis of rotation that is often far away from the vehicle. A vehicle tracker was constructed based upon this constraint and using the concepts presented in [9]. The local translation-based vehicle tracking system architecture is shown in Figure 5.

A vehicle is subdivided into local regions and each region is treated as if it had undergone purely translational motion. The extraction and grouping of features can be done automatically, but is more efficient and reliable when directed by a vehicle model. Instantiation of the vehicle model shown in Figure 4 spawns the local translation tracker. This tracker consists of feature extraction and grouping, and 3D trajectory computation via local translational decomposition and edge matching. The information derived from
this tracker can be used to determine a 3D road model, as well as refine the attributes of the instantiated vehicle model through temporal modeling with the Kalman Filter.

3.1 Feature Extraction

The local translation tracker requires features which can be matched in successive images. The type of features used are conventional masks of image pixels, extracted from distinct areas of the image. In the examples shown in this paper, the masks are 9x9 pixel arrays. Normalized correlation is used to determine similarity of extracted features. Normalized correlation is used both in measuring feature distinctiveness, and for evaluating the matches of extracted features along the radial flow determined by a possible axis of translation. Since the radial flow lines do not necessarily pass through the center of the image pixel arrays, bilinear interpolation is used to allow correlation and mask extraction at a continuous range of locations.

The distinctiveness of a feature is one minus the best correlation value obtained when the feature is correlated with its immediately neighboring areas. Good features are selected by finding the local maxima in the values of the distinctiveness measure over an image. This method of feature extraction is known as the Moravec interest operator. Neighborhoods over which the features are selected are constrained to areas that contain large intensity discontinuities, determined by extracting zero-crossings. This generally results in the extraction of areas of high curvature along the zero-crossing contours. As a vehicle is tracked over a sequence of images, the feature extraction process is continually reapplied. New features are the result of occlusions or changes in observable detail as a vehicle moves in depth.
3.2 Feature Extraction from a Model

The feature positions used to derive the local translation vectors are not direct measurements of the vehicle position. It is necessary to measure quantities in the image that are directly related to the position of the vehicle for the purpose of initializing the vehicle position, and measuring the quality of subsequently estimated positions. Searching for vehicle features such as headlights or wheels is one way to measure the position of the vehicle. Currently, the edges of the vehicle model provide the necessary measurements.

A sub-image is constructed about each edge that is to contribute to the measurement. The size of this sub-image is determined using the covariance matrices associated with the Kalman Filter (see Section 4). The sub-image is then convolved with a one dimensional edge mask oriented in the direction of the edge. Each row of the resulting sub-image is then summed, and the maximum values are considered possible positions for the edge. The position of the vehicle can be determined given three edge positions. The error sums of additional edges are added to the error sum for each possible position, and the maximum value is chosen as the position of the vehicle. An image with a user-instantiated vehicle model is shown in Figure 6. The edge-based position adjustment discussed above was applied to the image at the user-instantiated vehicle model position. The resulting vehicle model position is shown in Figure 7.

The extraction of pixel masks for local translation estimation is also controlled by the vehicle model. The vehicle model is used to determine the appropriate density of features at each image position. Figure 8 shows a set of features extracted from an image with a vehicle model. Notice that the density of features is much higher within the vehicle model where the features are used to calculate several 3D displacements. Outside the vehicle the features are used to calculate a single 2D displacement in order to register sequential images, and so fewer features are necessary.
Figure 6: User-instantiated vehicle model

Figure 7: Adjusted vehicle model position
Figure 8: Features grouped using a vehicle model

The vehicle model is also used to group the features lying on the vehicle. The local translation tracker computes a direction of translation at different points along the surface of the vehicle. Vehicle motion is locally planar, so the majority of rotation will be about the normal to this plane. Therefore, features that lie close together with respect to the length of the vehicle have similar trajectories. The local translation tracker groups features using this distance criterion. The results presented in this paper were all produced by grouping the vehicle features into three groups: the front, center, and rear features. These groups are shown in Figure 8.

The vehicle model is also used to determine areas in which feature extraction should be performed due to occlusion. Areas in the background are occluded by the vehicle, and as the vehicle moves these areas become visible and feature extraction is performed. Self-occlusion by the vehicle is also detected using the vehicle model. In this case, vehicle rotation exposes previously occluded areas on the vehicle.
4 Temporal Filtering

The problem of fusing multiple data measurements is one which arises frequently in the field of robotics and computer vision. The Kalman Filter is a tool which is capable of fusing multiple erroneous measurements while reducing the uncertainty associated with these measurements [1, 4, 8, 12, 18]. The Kalman Filter can be used to improve estimates of an object's size, position, velocity, and acceleration, and at the same time provide a statistically based measure of the uncertainty associated with these object parameters.

If a priori knowledge of a scene exists, this knowledge should be incorporated within the Kalman Filter to produce more reliable motion estimation techniques. Combining the Kalman Filter with an object model can result in a powerful tool for object tracking and motion estimation [5]. The object model provides constraints that can be incorporated within the Kalman Filter models. In this section a motion model is developed that will aid in the tracking of vehicles, this model exploits characteristics common to all vehicles providing a powerful yet general tool for vehicle tracking.

4.1 Kinematic Model

Vehicle motion is typically planar in a local neighborhood. Restricting the vehicle model to locally planar motion results in a filter with fewer parameters. The position of the vehicle is represented by a 2D vector $p$. The linear velocity is assumed to lie in the direction of the vehicle orientation, so only the magnitude of the velocity $v$ is needed. The orientation is represented by an angle $\phi$, and the angular velocity is also a single parameter $\omega$. The linear and angular acceleration of the vehicle are assumed to be zero. Any acceleration that occurs is incorporated in the Kalman Filter through the noise models of $v$ and $\omega$.

The dynamic vehicle model consists of five parameters $\mathbf{x} = [p_x, p_y, v, \phi, \omega]^T$. The
derivative of $x$ with respect to time is
\[
\frac{dx}{dt} = [v \cos \theta, v \sin \theta, 0, \omega, 0]^T
\]  
(1)

Integrating this equation results in the nonlinear system state equation
\[
f(x_i) = x_{i+1} = x_i + \begin{bmatrix}
v_i \cdot \sin(\theta_i + \omega_i \tau) - \sin \theta_i \\
v_i \cdot \cos(\theta_i + \omega_i \tau) - \cos \theta_i \\
0 \\
\omega_i \tau \\
0
\end{bmatrix}
\]  
(2)

where $\tau$ is the time between consecutive image frames. In order to apply the Kalman Filter equations to this problem, the nonlinear model $f(x_i)$ needs to be linearized. $f(x_i)$ can be linearized about the current state estimate $\hat{x}_i$ by a first order Taylor expansion
\[
f(x_i) \approx f(\hat{x}_i) + \frac{\partial f(\hat{x}_i)}{\partial x_i} (x_i - \hat{x}_i)
\]  
(3)

This equation can be rewritten as
\[
x_{i+1} = \Phi_{i+1,i} x_i + w_i
\]

where
\[
\Phi_{i+1,i} = \frac{\partial f(\hat{x}_i)}{\partial x_i}
\]
\[
w_i = f(\hat{x}_i) - \frac{\partial f(\hat{x}_i)}{\partial x_i} \hat{x}_i
\]

The predicted estimate is defined as
\[
\hat{x}_{i+1,i} = f(\hat{x}_i)
\]  
(4)

and the linear plant model $\Phi_{i+1,i}$ is the Jacobian matrix for $f$
\[
\Phi_{i+1,i} = \frac{\partial f(\hat{x}_i)}{\partial x_i} = \begin{bmatrix}
1 & 0 & s_i & v_i c_i & \frac{v_i}{\omega_i} \cdot [\cos(\theta_i + \omega_i \tau) \tau - s_i] \\
0 & 1 & -c_i & v_i s_i & \frac{v_i}{\omega_i} \cdot [\sin(\theta_i + \omega_i \tau) \tau + c_i] \\
0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 1 & \tau \\
0 & 0 & 0 & 0 & 1
\end{bmatrix}
\]  
(5)

14
where $s_i$ and $c_i$ are defined as

$$s_i = \frac{\sin(\theta_i + \omega_i \tau) - \sin \theta_i}{\omega_i}$$

$$c_i = \frac{\cos(\theta_i + \omega_i \tau) - \cos \theta_i}{\omega_i}$$

The model presented in this section assumes that the motion is in a ground plane perpendicular to the direction of gravity. However, errors in the direction of gravity, as well as other modeling errors, result in motion that is not purely planar with respect to the instantiated gravity direction. In order to allow the vehicle to deviate from the plane, an additional parameter was added to the system state model. This parameter is the distance of the vehicle above the plane of motion. The height parameter $h$ is independent of the system state vector $x$, so it is maintained separately from these other parameters. An assumption that $h$ remains constant over time was used along with an associated standard deviation $\sigma_h$ in order to construct a recursive mean square error estimator for $h$.

The kinematic parameters are initially assigned values obtained after the user positions the vehicle model and the edge-based adjustment has refined this initial position. The position $[p_x, p_y, h]$ and orientation $\phi$ are inferred directly from the position and orientation of the vehicle model. The velocity parameters $v$ and $\omega$ are derived by finding the position $[p_x, p_y]$ and orientation $\phi$ of the vehicle in an additional image frame. Measurements of $[p_x, p_y, h]$ are obtained using the edge-based adjustment discussed in Section 3.2, and measurements of $[v, \phi, \omega]$ are obtained using the local translation vectors.

The location of the vehicle at time $i + 1$ is predicted using the kinematic estimate provided by the Kalman Filter at time $i$. This predicted position is used to extract edge information and perform the edge-based adjustment in image frame $i + 1$. The adjusted vehicle position is then passed to the filter as the current position measurement. Local
translational decomposition is used to derive the remaining kinematic parameters. Features extracted from a vehicle are divided into three groups based upon spatial position as shown in Figure 8. Each group is treated as a translating rigid body, and the local translation vector is calculated for this section of the vehicle. A weighted average of the three local translation vectors is taken as an estimate of the linear velocity \( v \), the vehicle orientation \( \phi \), and the angular velocity \( \omega \). The rear axis of the vehicle is located between the middle and rear feature groups. The vehicle rotates about the rear axis and the vehicle motion at this point corresponds with the vehicle orientation. Therefore, the linear velocity is obtained by averaging the magnitudes of the middle and rear local translation vectors

\[
v = \frac{\| t_{\text{middle}} \| + \| t_{\text{rear}} \|}{2}
\]

and the vehicle orientation and angular velocity is obtained by averaging the three local translation vector orientations

\[
\phi = \frac{\theta_{\text{front}} + 2 \cdot \theta_{\text{middle}} + 5 \cdot \theta_{\text{rear}}}{8}
\]

\[
\omega = \frac{v}{d} \cdot (\theta_{\text{front}} - \theta_{\text{rear}})
\]

where \( d \) is the distance in 3D space between the front and rear feature groups. Using the information obtained from both the edge-based adjustment and the local translational decomposition a five dimensional measurement vector \( z = [p_x, p_y, v, \phi, \omega]^T \) is constructed. The measurement vector \( z \) is identical to the system state vector \( x \).

4.2 Error Models

Error models are maintained over time by the Kalman Filter. As more images become available, the filter updates the vehicle model and associated error models. Errors are modeled by the Kalman Filter through the covariance matrices \( P_t, Q_t, \) and \( R_t \). \( R_t \) is a model of the noise in the measurement equation. The error measure calculated when
performing the edge-based adjustment in Section 3.2 is scaled between one and ten and used to evaluate the quality of each measurement. This weighted measure \((w_p, w_v)\) is calculated for both the position estimate and the velocity estimate. The measurement noise model must also reflect the fact that the distance of the vehicle from the camera affects the reliability of the image-based measurements. A two pixel error at the image plane scaled by the distance of the vehicle along the optical axis is defined as \(e = \frac{2z}{f}\) where \(z\) is the distance of the vehicle from the camera and \(f\) is the camera focal length. Errors in the parameters \(p_y\) and \(h\) are dependent upon the direction of gravity \(g\) with respect to the optical axis of the camera, so the gravity direction was also considered. Using the terms \(e, w_p, w_v,\) and \(g\), the covariance matrix \(R_i\) was defined as a diagonal matrix with variances along the diagonal expressed as

\[
\sigma^2_{R_i} = \begin{bmatrix}
w_p e^2 \text{ pixels}^2 \\
w_p e^2(9g_y + 1) \text{ pixels}^2 \\
100w_v e^2 \text{ (pixels)}^2 \\
.003w_v \text{ radians}^2 \\
.03w_v \text{ radians}^2 \text{ (second)}^2
\end{bmatrix}
\]

The variance of the height parameter \(h\) was determined in a similar manner \(\sigma^2_h = w_p e^2(9g_z + 1) \text{ pixels}^2\). The linear and angular velocities were given large values in order to compensate for the vehicle accelerations which were not modeled.

The other covariance matrices were expressed in terms of \(R_i\). \(Q_i\) is a model of the noise in the system state equation. This covariance matrix is constant over time \((Q_i = Q_1)\) and was determined empirically to be \(Q_1 = \frac{R_1}{10}\) with \(w_p = w_v = 1\). \(P_i\) is a measure of the error in the estimate \(\hat{x}_i\) and is maintained over time by the Kalman Filter. \(P_i\) is set equal to the noise in the initial measurement \(R_1\), since the initial estimate \(\hat{x}_1\) is determined entirely from the image information.

\(P_i\) is used within the Kalman Filter for determining the amount of weight to be placed upon new estimates of the system state. In addition, \(P_i\) is used to determine the maximum allowable edge-based position adjustment in 3D space of the vehicle model (see
Section 3.2). As the estimate of the vehicle is refined, the search for the vehicle position can be restricted to a smaller area. The position adjustment algorithm is restricted using the standard deviation values associated with the position parameters $p_x$, $p_y$, and $h$. The maximum allowable adjustment in 3D space is defined as $3 \cdot \sqrt{\sigma_{p_x}^2 + \sigma_{p_y}^2 + \sigma_h^2}$. The adjustment perpendicular to the plane of motion is further restricted by defining a maximum allowable perpendicular adjustment of $3 \cdot \sigma_h$.

5 Results

The model-based vehicle tracking system described in this paper was tested on several image sequences. The sequences were shot from hand-held cameras and contained turning vehicles. Sequences which contained turning vehicles were chosen because rotational motion is the most difficult case of motion for the local translation tracker. The cameras were uncalibrated, so the camera centers and focal lengths were unknown. The optical axis was assumed to pierce the cameras at the center of each image. The focal lengths were set by the user prior to the tracking process.

5.1 Sequence #1

The first image sequence contained forty-five 512x480 images. For the results shown in this section a focal length of 950 pixels was used (field of view of 30.2 degrees). Initially a vehicle model and a gravity model were instantiated by a user. The position of the gravity model is shown in Figures 2 and 3. The initial vehicle position is shown in Figure 6. The edge-based adjustment discussed in Section 3.2 was then applied to the image at the user-instantiated vehicle model position, resulting in the position shown in Figure 7. Using the vehicle model to control the extraction and group the results, features were extracted as discussed in Section 3.2. Four groups of features were formed representing the front, middle, and rear of the vehicle as well as background features.
Figure 9: Local translation vectors for the three feature groups

The extracted and grouped features are shown in Figure 8. The background features were used to register the images by assuming that their motion was the result of translational motion parallel to the image plane.

Once the camera motion between frames 1 and 2 was calculated, a translational motion assumption was used to estimate the motion of the three groups of vehicle features. The search for an axis of translation was constrained by the direction of gravity and the current orientation of the vehicle. The plane perpendicular to the direction of gravity was searched in an area near the current vehicle orientation for the three local translation vectors. The results of this first search are shown in Figure 9. The top sphere contains the vector corresponding to the features on the front of the vehicle, the middle sphere corresponds to the middle of the vehicle, and the bottom sphere corresponds to the rear of the vehicle.

The three local translation vectors were used to calculate a new vehicle model position
and orientation. The linear and angular velocities were also derived, and the Kalman Filter model was initialized using the results. This process was repeated for image frames 2 and 3, except that the local translation vectors were used to calculate only the orientation and velocity parameters. The location of the vehicle at time 3 was predicted using the system state estimate provided by the Kalman Filter at time 2. This predicted position was used to extract edge information and perform the edge-based adjustment in image frame 3. The resulting position estimate was passed to the Kalman Filter along with the orientation and velocities derived from the local translation vectors. This process was repeated for the remaining image frames in the sequence. Figures 10-13 show the position of the vehicle model at image frames 10, 20, 30, and 40 respectively.

The actual vehicle length was unknown, so a value of 4 meters was chosen in order to calculate motion statistics. The remaining system parameters were then scaled relative to the truck length. Figure 14 shows the forty-five vehicle positions projected onto the plane perpendicular to the direction of gravity. The vehicle starts in the upper left hand corner of the plot and moves to the lower right corner. The height of the vehicle
Figure 11: Vehicle model and features for image frame 20

Figure 12: Vehicle model and features for image frame 30
Figure 13: Vehicle model and features for image frame 40

Figure 14: Vehicle trajectory for the 45 frame sequence
above the gravity plane is shown in Figure 15. This figure shows that the vehicle is moving slowly away from the plane of gravity. This is due to errors in the focal length, vehicle model, and the user-instantiated gravity plane. The height information shown in Figure 15 is presented to the user, so that the appropriate action can be taken. The user can then adjust any number of these parameters in an attempt to correct for the height error. The vehicle orientation \( \phi \) is shown in Figure 16. The orientation is measured with respect to the axis obtained by intersecting the image plane with the gravity plane. This axis corresponds to the x-axis shown in Figure 14. Finally, the linear and angular velocities of the vehicle for the forty-five frame sequence are shown in Figures 17 and 18.
Figure 17: Linear velocity of the vehicle

Figure 18: Angular velocity of the vehicle
5.2 Sequence #2

The second image sequence contained forty 320x240 images. Motion recovery from this sequence was difficult for several reasons. One reason was that the size of the vehicle in the images was very small, occupying approximately a 40x20 pixel area. Another reason was that the vehicle trajectory was not planar. Finally, the vehicle was accelerating throughout the sequence. These last two conditions were difficult because the motion model used in this tracker assumes planar motion with no acceleration.

The relative scale of the truck model used in the first sequence was modified to better fit the car that was tracked in the second sequence. Figure 19 shows the model used for the second sequence. The focal length of the camera was set to 600 pixels (field of view of 29.9 degrees). Initially a vehicle model and a gravity model were instantiated by a user. The position of the gravity model is shown in Figures 20 and 21, and the initial
vehicle model position is shown in Figure 22. The edge-based position adjustment was then applied to the image at the user-instantiated vehicle model position, resulting in the position shown in Figure 23. This initial vehicle model position was used to extract and group features. Figure 24 shows the features extracted from the first image frame. Using the algorithms discussed in this paper, the vehicle was tracked throughout the forty frame sequence. Figures 25-28 show the vehicle model position in image frames 10, 20, 30, and 40 respectively.
Figure 23: Adjusted vehicle model for image frame 1

Figure 24: Extracted and grouped features for image frame 1
Figure 25: Vehicle model position at image frame 10

Figure 26: Vehicle model position at image frame 20
Figure 27: Vehicle model position at image frame 30

Figure 28: Vehicle model position at image frame 40
Figure 29: Vehicle trajectory for the 40 frame sequence

Figure 30: Height of the vehicle above the gravity plane

The actual vehicle length was unknown, so a value of 3.5 meters was chosen in order to calculate motion statistics. The remaining system parameters were then scaled relative to the length of the car. Figure 29 shows the forty vehicle positions projected onto the plane perpendicular to the direction of gravity. The vehicle starts on the right side of the plot and moves to the upper left corner. The height of the vehicle above the gravity plane is shown in Figure 30. Figure 31 shows the vehicle orientation $\phi$, which is measured with respect to the axis obtained by intersecting the image plane with the gravity plane. This axis corresponds to the x-axis in Figure 29. Finally, the linear and angular velocities of the vehicle for the forty frame sequence are shown in Figures 32 and 33.
Figure 31: Vehicle orientation

Figure 32: Linear velocity of the vehicle

Figure 33: Angular velocity of the vehicle
6 Conclusion

This paper introduced an interactive model-based system for vehicle tracking. The tracker makes use of user-instantiated gravity and vehicle models to track vehicles in image sequences shot with hand-held uncalibrated cameras. The results presented in this paper show the feasibility of this interactive model-based technique for constructing simple yet robust solutions to difficult image understanding problems.

There are several areas in which this work can be extended. The current user interface consists of a pull-down menu and a mouse. The models are instantiated by choosing items from the menu, the user can then manipulate the objects by using the mouse. The introduction of more sophisticated interactive devices such as a data glove would allow the development of a more intuitive interface. Users could grab objects and place them into the scene quickly and accurately. Another limitation of the current system is that the models are manipulated and projected onto a static image. A real-time system would necessitate users instantiating the models in a dynamic scene. This type of instantiation will present new complexities, but also result in additional opportunities for exploiting the user input.

Another area that warrants future research is in the development of models. Perceptual processing such as segmentation which is sensitive to the material properties of objects would result in more robust objects. The method of combining constraints imposed by instantiated objects is another area that needs to be investigated. Currently inter-object constraints are represented implicitly within each of the object models. This is reasonable when dealing with a small number of models, but as the number and complexity of models grow there is a need to represent and maintain these constraints explicitly. One way this could be accomplished is through the use of an existing constraint satisfaction algorithm such as DeltaBlue [14].
References


