Surveillance – Enhanced Collision Avoidance

Project Proposal

Rohit Annigeri, Akash Joshi, Yifei Shen, Thaddaeus Voss

February 17, 2017

**1 Abstract**

Recently, more and more technology companies are interested in developing smart, autonomous cars. Using sensors installed around the car body, the smart car can get information from its surroundings and drive itself. This method works well on highways and roads since the car follows the flow of traffic, and generally, the sensor can gather this information under many environmental conditions. However, when crossing the intersection, the situation becomes more complicated. Vehicle sensors might be blocked by other cars and objects and may fail to get information of cars moving in the other direction. These scenarios can be dangerous for autonomous cars. Our group tries to use the surveillance camera, which is widely used on intersections, to predict the collision and send information to the cars to compensate for sensors dead zones and avoid accident.

**2 Description of the system**

The basic parts of our system are shown in the block diagram below:

First the surveillance camera will monitor the intersection and we will track the cars passing through the intersection. Next the velocity and moving direction of these cars will be calculated. Based on this information, the likelihood of a collision can be calculated and updated with time. Finally, the control information will be sent to cars to change their course and avoid the accident.

This system can be integrated with the traffic monitors that already exist in intersections and provide addition information for autonomous car and even non-autonomous driving cars.

The prototype will include the analysis system applied to live CCTV or prerecorded videos, and a simulation will be conducted in lab using a simple camera as the surveillance camera and remote-controlled cars to simulate the real cars.

Flow chart on following page

Surveillance camera

Real time video of intersection

Track the cars in the intersection

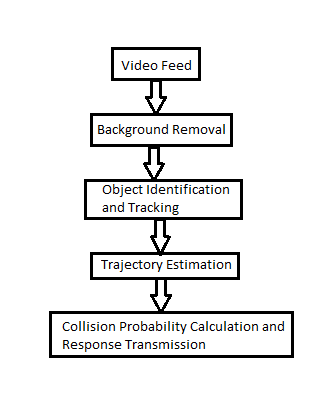
Get cars’ velocity, moving direction and other information

Predict the possibility of collision

Send messages to cars and control their move when necessary to avoid collision

**3 Description of Possible Algorithms**

Algorithmic Flow Chart:



The algorithmic system consists of two major parts, one being the tracking of multiple objects on a real time video feed and the second being estimating the velocities and possible direction of the object. The possible algorithms will be evaluated on the basis of execution time, resulting accuracy of the video quality and content.

Motion based object tracking can be divided into two parts[1].

1. Detecting significant objects in each frame.

2. Associating motion across frames to each object.

To detect significant objects in each frame, first the background model must be estimated. The following algorithms can be considered[2][3]:

1. Frame Difference: Subtracting consecutive frames to eliminate the background. One disadvantage here is the assumption that all foreground objects are moving and background is static.

2. Temporal Median Filter: This uses the previous frames to calculate a median for each pixel as threshold and construct a reference background model. One disadvantage is high computational cost to store previous pixel information.

3. Mean Filter: Calculates the mean of each pixel using the previous frames to estimate a threshold which distinguishes between background and foreground objects. Disadvantages can stem from using a global or time independent thresholds.

4. Running Gaussian Average: A Gaussian probability density function is modeled to each pixel based on the previous frames. The mean and variance are updated on each frame. The threshold is estimated based on the difference between the pixel value and it’s mean scaled by the standard deviation. Advantages are lower computational cost and ability to handle soft illumination changes.

5. Mixed Gaussian Models: Uses multiple Gaussian models for every pixel using each of the color channels. Robust to soft illumination changes. Complex model to implement.

After modeling the background, this is subtracted from the current frame to get the foreground objects. To compensate for spurious artifacts, morphological processing (opening or closing) is applied[4].

Connected component labeling could be used to identify individual objects. Blob detection based on color, shape or size can be used to detect moving objects and possibly distinguish between objects. A small neural network could also be trained to identify between objects[5][6].

Tracking objects across frames involves identifying accurately the object in the new frame. This process can be expedited by using a local search window for each object within which the object could appear with high likelihood. To identify accurately the following methods can be applied in each of the search windows.

1. Correlation based block matching: Given a region in one frame, a corresponding region in next frame with highest correlation is found within the search window. This is computationally costly.

2. Feature based matching: Detect features on each object and track them between each frame. This also includes shape, color, texture, interest point detectors and invariant local descriptors. Feature detection faces weaknesses such as feature disappearance as car orientation changes, car is partially visible, drift error etc.

3. Mean Shift Tracking: This tracks objects by using their histogram as a feature vector. It generates a confidence map in the neighborhood of the object and finds the location with the highest probability. Accuracy can be improved by considering a color video. However the local search window size selection is not trivial and depends on object displacement[7].

4. Kalman Filtering: A Kalman filter has to be applied to every object to track trajectories. The Kalman filter is used to predict the location of the object in the next frame. To supplement in the correction feedback to the Kalman filter, correlation based block matching could be used in the neighborhood of the predicted location.

To get trajectory information, homographic transformation needs to be done in order to convert image coordinates into real world coordinates[8]. This would require trial and error and case by case analysis to have accurate conversions. Using this transformation in unison with the predicted locations from Kalman filter and frame rate, velocity, direction and displacement vectors are found. This information is used the second stage to predict crash probabilities.

To quantify crash predictions, we would look into our own in-house method. Each object’s trajectories is converted to linear equations and solved to check if collision is possible or not. At the point of collision every object (here car) is given bounding box and the area of overlap between the these boxes could be used as a probability metric.

**4 Complexity Analysis**

The system consists primarily of two parts.

1. Control center: This acts as the master junction where the DSP is fed a real time video and responds with a RF transmitter if a collision is predicted.

2. Local environment: Consists of an intersection and independently moving cars with RF receivers

Bottlenecks:

1. The accuracy of detection of cars depends on position of camera and its distance from the car. If the car appears too small in the video it may not be detected as a significant object.

2. The homographic transformation may be inaccurate, leading to poor position and velocity estimates. This in turn could effect crash prediction.

3. The car reacting appropriately to the response signal could be depend on external factors such braking distance of the car.

**5 Major Challenges**

1. Accurately estimating the background, in case of a changing background (illumination) or an object stopping. Possible solution is using a Gaussian running average modified by not accepting tracked objects which come to a stop[2][3].

2. Perspective transformation and estimating trajectories. Inaccuracies in perspective could snowball into poor crash predictions. Multiple trials would be required to improve transformation results.

3. Quantifying results as probability and setting it as an appropriate threshold. Our own in house algorithm needs to established theoretically before applying to physical models.

**6 Modeling Tasks**

The task ahead of us is to now model several vehicle collision scenarios. There are some aspects of the model that are fixed such as, we always consider vehicles travelling straight on a highway or meet near an intersection. We also consider that the model is based on collision of two vehicles and not with a non-vehicle obstacle. The next phase is creating this model will be based on developing a hardware system using two autonomously driven vehicles which run on a predefined collision course. The RC cars will be made to follow a certain path and depending the type of collision required an algorithm will be developed to create allow the two RC cars to follow a predefined path of collision [9].

There are many collision courses that can be considered during modeling a collision scenario such as, forward collision, merging collision, changing lanes collision or collision at an intersection. This predefined collision paths with be modeled on a RC car and with the help of sensors and cameras the collision avoidance system will be able to take preventive measures.

**7 Human Factors**

While our project and its final test setup concerns the control of cars using no human input or interface, we would like to include the human evaluation of the warnings it sends to cars in near-collision and collision scenarios. We will ask classmates and others to evaluate the frequency, content, and delivery of warnings to the cars in these scenarios to determine if the warnings and control of cars is appropriate in several scenarios.

**8 Training**

This project is based on real time video. We will use real, archived surveillance videos which contain both collision and normal traffic flow as testing data. These videos can be found online. What is more we will also use MATLAB to simulate the intersection traffic and also build a simulated crossing with remote controlled cars and camera.

**9 Rough schedule**

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Milestone** | **Associated Tasks** | **Completion Date** |
| Milestone 1 | Calculate speed, direction vector, and displacement of objects in a sample intersection surveillance video | * Pre-process input video * Isolate moving objects from background * Apply Kalman filtering to calculate metrics * Overlay data on the input video / MATLAB simulation * Test algorithms on live feed data | March 13 |
| Milestone 2 | Successfully run intersection simulation using remote-controlled cars | * Obtain two remote-controlled cars * Develop code to control two cars using relevant hardware * Perform tests controlling cars at different speeds, in collision and non-collision scenarios | April 1 |
| Milestone 3 | Derive percentage likelihood of a crash using car and human movement data from video feed and apply these warnings to intersection simulation | * Process mobile object metrics in real-time * Develop a set of equations to predict percentage likelihood of a crash using vectors and movement data * Continually test these algorithms on test vectors, sample intersection surveillance data, and the running simulation * Overlay these warnings live onto the video feed and evaluate human responses to warnings | April 15 |

**10 Final Test Set-Up**

For the demonstration of our full project, we will use the following equipment:

* + 1 TI TMS320DM6437 Fixed-Point DSP Board
  + 1 Overhead Video Camera
  + 2 Remote-Controlled Cars
  + 1 RF Transmission Board
  + 1 Physical Mini-Intersection for Simulation

To simulate our technology working in a real world simulation environment, we will create a mock intersection and run the remote-controlled cars on the intersection in different scenarios with the camera pointed at the intersection from above, with video being fed into the DSP. These scenarios include vehicle-vehicle and pedestrian-vehicle non-collision, near-collision, and collision scenarios. We will demonstrate warnings being overlaid onto the live feed of the mock intersection, and use our collision prediction algorithms to derive the likelihood of a collision. Based on this data, we will have the DSP send signals to an RF transmitter board that will control the vehicles. This represents the system being able to interact with autonomous vehicles and also the sending of warnings to non-autonomous vehicles.

**11 Board**

We will use the TI DSP Board TMS320DM6437 Fixed-Point for processing the video feed from a camera. This is the main point of processing for all of our project’s programming needs.

In addition to the TI DSP board, we will also use XBee communications boards to communicate between the DSP and two remote control cars over RF frequency. We will observe the cars and calculate their movement metrics from a video feed out of the camera connected to the TI DSP board.

**References**

[1] T. MathWorks, "Motion-based multiple object tracking," 1994. [Online]. Available: https://www.mathworks.com/help/vision/examples/motion-based-multiple-object-tracking.html?requestedDomain=www.mathworks.com. Accessed: Feb. 14, 2017.

[2]A. Sobral and A. Vacavant, "A comprehensive review of background subtraction algorithms evaluated with synthetic and real videos," *Computer Vision and Image Understanding*, vol. 122, pp. 4–21, May 2014.

[3] J. Gallego, M. Pardàs, and G. Haro, "Enhanced foreground segmentation and tracking combining Bayesian background, shadow and foreground modeling," *Pattern Recognition Letters*, vol. 33, no. 12, pp. 1558–1568, Sep. 2012.

[4] R. Andrews, "Tracking Multiple Objects in Real Time," Oct. 1999.

[5] B. Coifman, D. Beymer, P. McLauchlan, and J. Malik, "A real-time computer vision system for vehicle tracking and traffic surveillance," *Transportation Research Part C: Emerging Technologies*, vol. 6, no. 4, pp. 271–288, Aug. 1998.

[6] [Online]. Available: https://pdfs.semanticscholar.org/0674/792f5edac72b77fb1297572c15b153576418.pdf. Accessed: Feb. 14, 2017.

[7] [Online]. Available: http://www.cse.psu.edu/~rtc12/CSE598G/introMeanShift\_6pp.pdf. Accessed: Feb. 14, 2017.

[8] S. Kang, J. Paikab, A. Koschana, B. Abidia, and M. Abidia, "Real-time video tracking using PTZ cameras," May 2003.

[9] [Online]. Available:

https://www.isr.umd.edu/~austin/enes489p/projects2011a/CollisionAvoidance-