**CS 4337**

**Final Project Report**

**DeerWatch:**

**A Computer Vision System for Mitigating Deer-Vehicle Collisions**

**Group 5 Members:**

* Ryan Lynch (ygd8)
* Jake Butler (nkf17)
* Samuel Manzanares (sam234)

**Problem Statement:**   
  
 Texas Parks and Wildlife estimates over 5 million deer inhabit the state. According to the Insurance Institute for Highway Safety (IIHS), Texas had the highest number of deaths from animal-vehicle collisions over a ten-year period from 2009-2018. According to a 2019 publication from the UT Department of Civil, Architectural and Environmental Engineering, “51,522 animal-related crashes were reported in Texas from 2010 through 2016, at a total cost

over $1.3 billion annually to Texas motorists.” The same paper reported that in insurance claims, “deer show up over 25 times more than the next animal, racoons.” On a larger scale, the IIHS reported more than 1.5 million deer-vehicle collisions occur each year across the U.S.

**Goal:**

We would like to design and implement a computer vision program which detects deer that are moving towards or around highways/roads. The program is meant to provide a starting point for a more cohesive application that warns drivers about deer that are actively crossing or present near the road ahead. If there are no deer detected, the warning signal is green. If deer are detected but not moving in the direction of the road, the signal is yellow. Lastly, if deer are present and moving in the direction of the road, the signal is red. Our program intends to be a proposition for mitigating deer-vehicle collisions, consequentially saving lives, money, and wildlife.

**Computer Vision Techniques Utilized:**

* Binary manipulation, thresholding, image morphology, motion detection using frame differencing, contour extraction, connected component analysis, measuring distance in 2D

**Methodology:**

Our program is split amongst five python files, four of which contain a primary function that is imported to main – where the program is demoed. Each file uses OpenCV to perform different computer vision techniques that contribute to the program as a whole:

1. **extract\_frames.py:**

* Purpose:
  + The ‘extract\_frames’ function is designed to extract frames from a given video file. It allows for selective frame extraction based on a specified frame rate, providing flexibility in how many frames are captured from the video.

* Parameters:
  + ‘video\_filepath’: The file path of the video from which frames are to be extracted
  + ‘frame\_rate’: Determines the frequency of frames to be extracted
* Functionality:
  + Once the video\_filepath is loaded using the ‘cv2.VideoCapture’ method, frame extraction enters a loop, extracting frames one by one from the video.

* Return:
  + Returns a list of frames (‘frames’). Each element in this list is a frame captured from the video, represented as an image array in the format used by OpenCV.

1. **get\_road\_lines.py:**

* Purpose:
  + The ‘get\_road\_lines” function is designed to identify and return the contours of the road in a given frame. This function focuses on detecting the edges or boundaries of the road by utilizing color segmentation and contour detection techniques. The contours of the road are then used for not only visual purposes, but are also used to identify the direction the deer are moving relative to the position of the road, thereby enabling the system to signal potential collision risks.
* Parameters:
  + ‘frame’: An image (frame) from a video in which the road lines or contours are to be detected.
* Functionality:
  + By data pre-processing in the HSV color space, color intensities of the road are determined. HSV was chosen so we could use OpenCV’s inRange function, which essentially allows us to threshold a particular inner range of pixel values, as opposed to just above a single value. A mask of the intensities within the established bounds is created, which is then fine-tuned using dilation, inversion, and slicing. Originally, we intended to use a Hough transform to determine the ‘boundary,’ but after some trial and error due to the curvature of the road, we decided creating a contour would be a more applicable approach. Contour’s were ideal since we didn’t want just a road line with a start and end point, but rather every single point in the line, which contour’s are perfect for as they are a vector of points in the image. This will be needed later when determining the direction the deer are moving.

* Return:
  + Returns the contours of the road as detected in the frame. These contours are represented as a list of points defining the boundary of each detected road segment.

1. **identify\_objects.py:**

* Purpose:
  + The ‘identify\_objects’ function is a key component in the system designed to monitor and signal deer movement near roads. It aims to identify deer that are moving in a sequence of video frames and returns their central positions. This function uses color mapping and motion detection to isolate moving deer in the frames. The information it provides is used in conjunction with road contour data to determine the proximity and direction of deer movement relative to the road. This is a key component of the system's signal mechanism.
* Parameters:
  + ‘past\_frame’, ‘current\_frame’, future\_frame’: These are consecutive frames from a video. The function analyzes changes between these frames to detect motion.
* Functionality:
  + To identify deer within the frame, the function starts by isolating brown colors in the frame using ‘make\_mask,’ then identifies motion through ‘get\_motion\_image’ using three-frame-differencing. The overlapping result of these two masks ensures only brown and moving objects are considered. The function then refines this detection using morphological operations and outlines the detected objects with ‘get\_rectangles’. It then calculates the centers of the largest objects using ‘rec\_area’ and ‘rec\_center’, and visually marks them with ‘draw\_rectangle’.
* Return:
  + Returns a list of coordinates(‘objects\_centers’), each representing the center of a detected moving object (deer).
* Helpers:
  + The ‘make\_mask’ function is responsible for isolating brown-colored areas in the frame, a step for identifying deer based on their fur color. It achieves this through HSV color space conversion and color thresholding.
  + The ‘get\_motion\_image’ function enhances the detection process by highlighting areas of motion between consecutive frames, using grayscale conversion and differential thresholding.
  + To delineate and analyze detected objects, the ‘get\_rectangle’ function utilized connected component analysis to find and bound objects within the frame.
  + To calculate the size and location of these objects, ‘rec\_area’ calculates the area of the bounding rectangles, and ‘rec\_center’ determines their central point. Lastly, ‘draw\_rectangle’ visually marks these objects on the frame for easy identification and analysis.

1. **get\_direction.py**

* Purpose:
  + The ‘get\_direction’ function determines whether the detected deer are moving closer to the road, staying stationary, or moving away from the road. This assessment is essential for generating appropriate warning signals to prevent collisions. By evaluating the direction of movement of detected deer relative to the road, the system can signal potential danger, thereby enhancing road safety.
* Parameters:
  + ‘prev\_object\_centers’: Coordinates of the centers of detected objects(deer) in the previous frame.
  + ‘object\_centers’: coordinates of the centers of detected objects(deer) in the current frame.
  + ‘road\_contours’: Contours of the road as detected in the current frame.
* Functionality:
  + The get\_direction module uses one helper function – get\_shortest\_distance() to establish whether or not the deer is moving towards the road. The function takes in the center of one of the ‘deer motions,’ and the contours of the road that we created. Then we were able to get the shortest distance between a deer and the road contour thanks to the pointPolygonTest function, which saved us testing every single point in the contour. The helper function is called for every entry in two lists of centers of motion, one list from the previous frame, and one from the current frame. To determine whether or not a deer is moving towards the road, the minimum of each list is taken, which in theory would be the deer closest to the road. Next, a ratio of the previous minimum over the current minimum is taken. If the current deer is at least 5% closer to the road than the previous deer, the function returns True, otherwise it returns False.
* Return:
  + A boolean value: ‘True’ if an object is detected moving closer to the road, ‘False’ otherwise
* Helpers:
  + ‘get\_direction’ utilizes the ‘get\_shorter\_distance’ helper function to evaluate the proximity of detected objects to the road. This function calculates the minimum distance between each object’s center and the nearest road contour, enabling ‘get\_direction’ to assess whether these objects are moving closer to, away from, or remaining stationary relative to the road.

1. **main.py**

* Main combines the modules used for deer detection and safety analysis. It loads a video, extracts its frames, and processes each frame sequentially. This processing involves detecting moving deer using ‘identify\_obejcts’, identifying road contours with ‘get\_road\_lines’, and assessing movement direction towards the road using ‘get\_direction’. Based on this assessment, it draws a signal to the top left of the video. The signal remains red when the program determines that deer are moving towards the road, yellow when the program determines deer are present but not moving towards the road, and green if the program determines no deer are present. The signal lingers on red to account for deer that might bolt across the road after one another. In addition to the bounding boxes drawn around deer, contours of the road are also drawn to each frame to provide a better visual for what the program is doing.

**Conclusion:**

In conclusion, our project attempts to utilize computer vision technologies as a starting point to mitigate deer vehicle collisions. By combining the CV technologies mentioned above our system detects and tracks deer near roadways and provides an alert that would theoretically help mitigate accidents. Throughout the project, we faced several challenges that included developing a reliable method to identify deer, accurately determining their direction of movement relative to the road, while reducing how computationally expensive the program would be. We chose to use a combination of motion detection and color masking in order to detect the deer. Both were chosen as just motion would mean the cars would also be detected, and just color would mean anything that is brown in the image, like some of the foliage, would be detected. By using both, we were able to only detect the deer. Our method of finding the deer’s movement direction was chosen as we found it would be a good combination of accuracy and performance. By condensing the deer down to just a single point, we drastically reduced the number of comparisons we needed to make.

The program certainly has room for improvement, however it archives what it aims to do. In its current version, we augmented the signal to err on the side of caution when it comes to detection. The video we initially considered ideal for developing our program proved to be more challenging than anticipated, as it contained dense clusters of deer. Nevertheless, the program serves as a solid foundation for future enhancements and iterations. It also helped us gain more experience using computer vision techniques, especially working with videos. This was a fun and engaging project to experiment with, and we expect to continue our work, or at least take what we learned and apply it to our future projects.

**Sources:**

<http://tx-aaa.iprsoftware.com/news/aaa-texas-reminds-drivers-most-automobile-crashes-involving-deer-happen-this-time-of-year>

<https://www.caee.utexas.edu/prof/kockelman/public_html/TRB19AVCs.pdf>

<https://www.catto.com/blog/2016-11-28/texas-leads-nation-vehicleanimal-collisions>

<https://www.dallascarwreck.com/personal-injury/car-accident/deer-car-accident/>