OptiBrake

Alex Asch, A69033251, aasch@ucsd.edu,
Benjamin Scott A16380204 bmscott@ucsd.edu

3. **Motivation:**

Recent developments in E-(pedal assisted) bicycles mean that more consumer bicycles incorporate battery and electric power. This also is leading to an increase in bicycles as commuter products, and thus we are seeing many mixed pedestrian and cyclist use areas, increasing odds of collision. Thus, by applying collision-detecting braking similar to automobiles, we want to reduce the rate of these collisions.

4. **Related Work**

Notable related work involves automated braking technology in automobiles, which is similar in idea, but often uses non-vision sensors like LIDAR and ultrasonic, while we mainly rely on a single digital camera. While it is true that image sensing using LIDAR would enable higher accuracy of capturing centimeter-level data for each camera frame, the high cost of the sensor led us to try a more affordable approach for the project using a digital camera instead and attempt to mitigate scaling issues through other multi-sensor information.

Related works using monocular vision systems have been researched and developed for Unmanned Aerial Vehicles (UAVs) and for robotics applications where environments are restriction free, i.e. the robot can move to whichever direction it finds suitable. These systems typically use optical flow which is the pattern of apparent motion of objects across successive video frames, which gives an approximation of the projection of the potential field in the 3D world to the image plane. This method assumes that when the camera approaches the obstacle, the obstacle's image in the frame grows, which causes the greater optical flow amplitude on the edges of the obstacle image. For most implementations, after locating the areas of the frame containing large pixel displacements, the collision system will be able to make a decision to perform. It's important to note that optical flow enables the algorithm to work with any type of obstacles, critical for our application to bicycle collisions.

Looking at the domain of UAVs specifically, one work used dense optical flow (Gunnar Farneback), which gives the flow vectors of the entire frame (all pixels) with high computational demand. While their UAV was successful in detecting and avoiding obstacles, the algorithm is not able to estimate the actual distances to the obstacles as it flies above ground features and focuses on the detection of obstacles without estimating the distance to obstacles. One of our goals was to find this true physical distance to collision metric using wheel odometry sensing to get more accurate results.

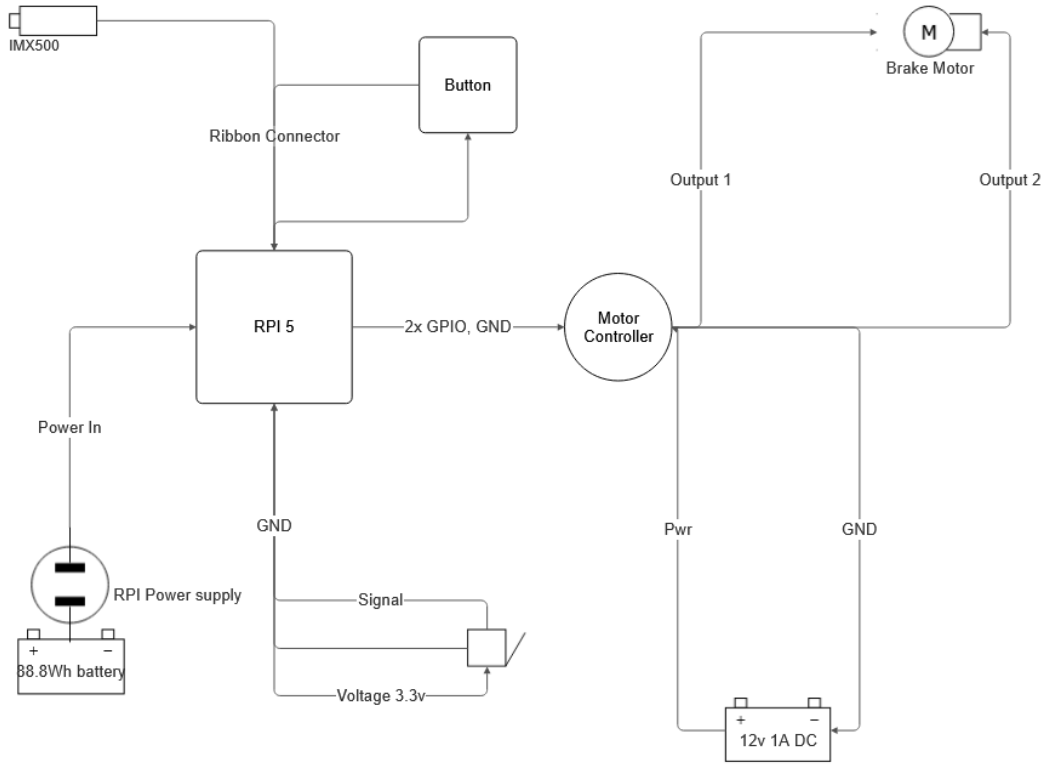
Another work uses a single front camera to estimate the distance to objects or a depth map for collision in UAVs using monocular distance estimation through a deep CNN (MegaDepth), but these 3-D constructions still suffer from not being scale-invariant or relative depth, leading to low accuracy applied in short ranges. Therefore, we choose not to pursue this approach.

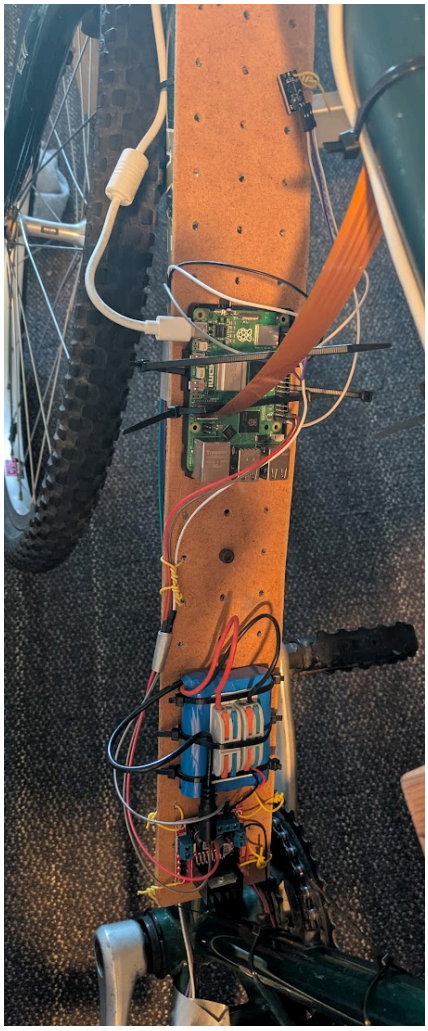
Also, since UAV flight control is more robust and stable to their environment, most UAV vision work assumes stable video between frames. Our implementation assumed this stabilization as well, but proved to be one of the key challenges, resulting in optical flow vector noise that caused a higher false positive rate.

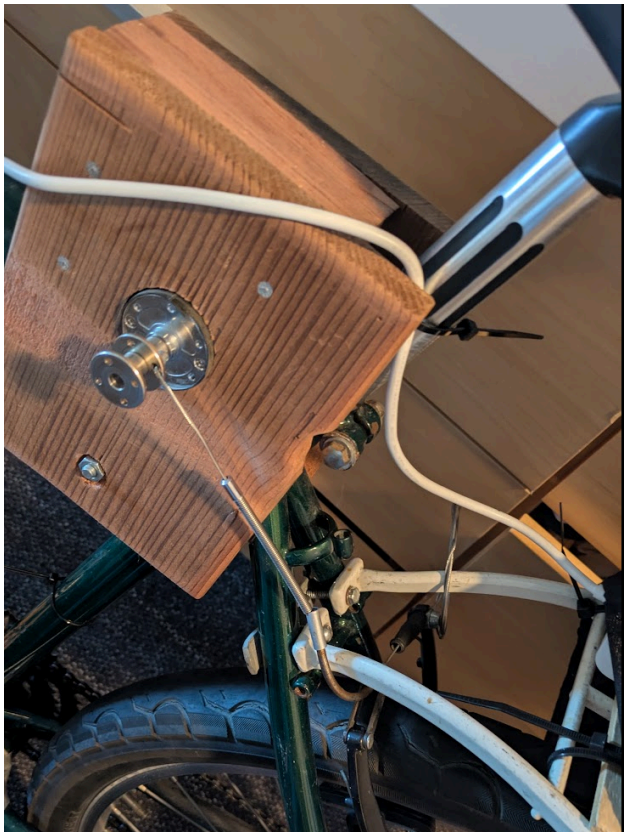
Our work is notable due to the vehicle we are mounting on. Having a less stable and more feature prone field of view leads to challenges in the braking system, as well as a more challenging environment to use optical flow in. With the digital camera in front of the bicycle, it is closer to the ground and therefore has many features to detect especially in crowded environments. Bicycles tend to be imbalanced at low speeds (jittery) which can directly cause hyper-sensitivity to surrounding features on its side. This instability and sensitivity proved to be the most challenging part of our implementation we tuned for from the computer vision perspective. The focus of this research was to implement a low-cost collision avoidance system using optical flow for bicycling applications at an average medium speed.

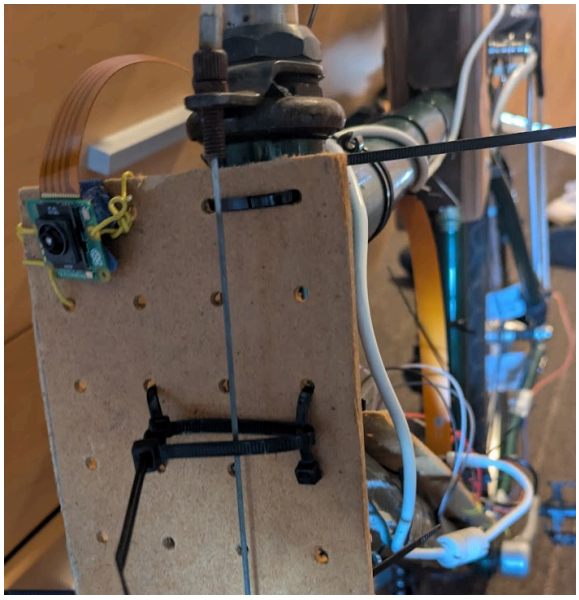
5.**Hardware Components**

- Raspberry Pi AI Camera| Sony IMX500
- Keyestudio collision sensor
- SinKeu 88.8Wh|65Watts Portable Laptop Charger
- KBT 12V 1200mAh Rechargeable Li-ion Battery
- Greartisan DC 12V 150RPM Gear Motor High Torque Electric Micro Speed Reduction
- WWZMDiB 2 Pcs L298N Motor Driver Controller Board DC Dual H Bridge Module
- Bicycle (Raleigh)
- V-Brakes (unknown manufacturer)
- Below are attached images of our hardware and wiring setup





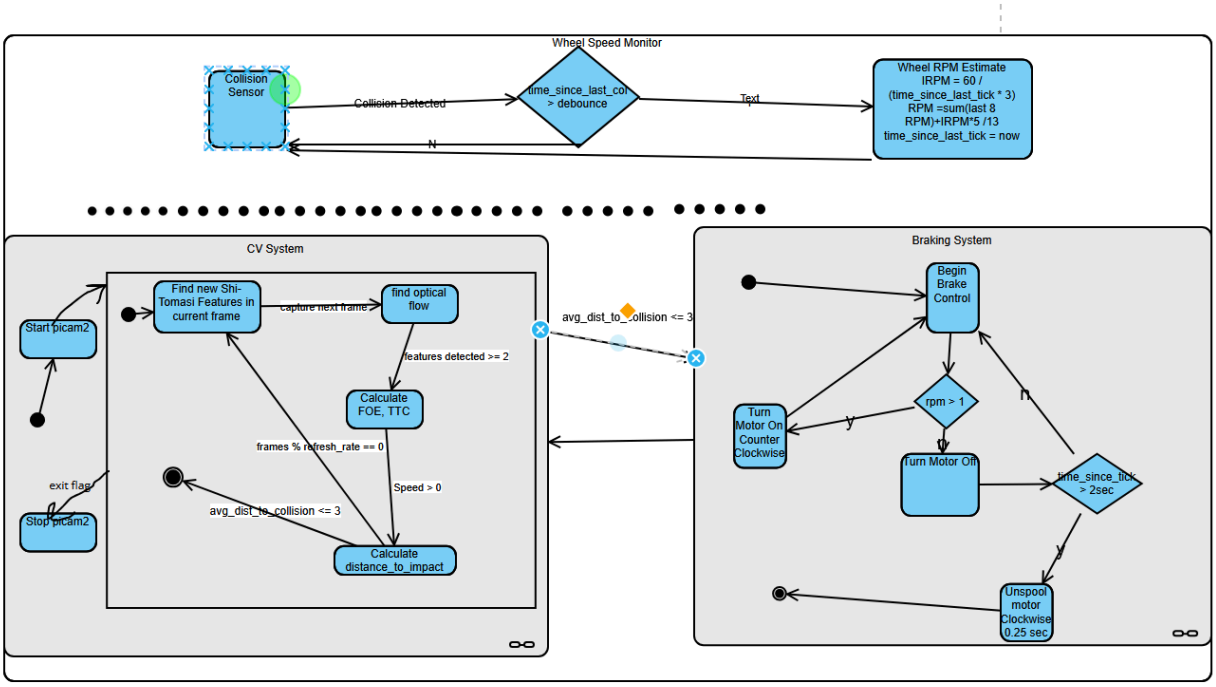




- We opted to use the AI accelerated camera, since we believed that we would be incorporating a neural network for object detection. We did not end up utilizing the accelerator for optical flow.
- We utilized a collision sensor to measure wheel speed in collision ticks from cards mounted on the spokes, since compared to IR based solutions, it was more reasonable in accuracy for our price point.
- We used two batteries, since our motor is 12v 1A, and our primary battery pack is 12v 1A out. Thus, we opted for one battery for the motor and one motor for the raspberry pi.
- We utilized a 150RPm motor, since that seemed to be the appropriate balance between torque and speed for our power draw. We did not have the budget to test multiple gearings of motor.
- We utilized V-Brakes, since cantilever brakes required too much force to actuate for our motor setup. More powerful "lighter" brake solutions such as disk brakes, and hydraulic systems were not within our budget to test.

6. **Software Components**

- Optical Flow Lucas-Kanade Method
- Focus of Expansion and TTC with Odometry of Wheel
- Calibration of Wheel Speed for improving reliability
- Gpiozero python library (2.0.1)



- We utilized the gpiozero python library, since it allowed us to work closely with our camera and computer vision software, while processing collision sensor ticks in an interrupt based manner. The relatively simple interface for GPIO devices was sufficient for our needs, and the interrupt based handling mechanisms were necessary for our collision sensor.
- We chose to use Lucas-Kanade optical flow instead of a neural network based implementation, since the main metric of importance is collision and proximity to any obstacles, and thus the recognition of objects in frame is not useful for our purpose.

7. For the braking sensor, we utilized an interrupt based wheel speed monitor using the collision sensor, and simply actuated the motor until the wheel speed (rpm), was detected at zero. After which, we waited a brief delay, and then rotated the motor in reverse, allowing us to unspool the braking cable. The amount to unspool was obtained via trial and error. While the above mechanism seems relatively simple, the collision sensor based wheel speed detector required heavy tuning. The first issue presented was to debounce the sensor, for each tick, the sensor read between 10-24 separate button press events, each a very small fraction of time apart. Thus, we utilized two time counters, once since the last registered tick, and once since the last real tick. If the time since the last real tick is greater than the debounce, we register it as a registered tick, and utilize only time between registered ticks to approximate wheel RPM as a function of ticks per rotation and time between ticks. We also were required to reduce the ticks per rotation from 9 to 3, to get a better discernment between a real tick and a debounced tick. The next challenge was the instantaneous timing measure between each tick to calculate rpm was a rather inaccurate measure of rpm, since our playing card and sensor mechanism was not physically exact to ⅓ of a rotation each tick. Thus, we calculated our RPM as a global weighted average between the past 9 RPM detections, with the current rpm metric weighted 5x as high for our average. This allowed a reasonable rpm and thus speed metric for control. From many repeated tests, we were able to appropriately debounce and tune these rpm and speed metrics to be reasonable values.

The attempted implementation of ABS was also not feasible. The first problem with testing and implementing an abs system was the motor power. Without a contrived set up, such as the rider changing the center of gravity off of the rear wheel, it was not possible to lock the rear wheel with our braking set up. However, even utilizing a contrived experimental setup, implementation of ABS proved to not be tenable. Our first challenge was with pulse width modulated DC motors. Since an intermittent rapid signal is registered by the motor as a pulse, the type of feathering and rapid fine tuned control of the braking mechanism as required for ABS was not possible with a DC motor and cable setup. Additionally, due to the short stopping distance of a bicycle, combined with our speed measuring sensor, it was not possible to obtain a rapid and accurate enough measurement to actuate braking control based on locking of the wheel, before the bicycle was fully stopped.

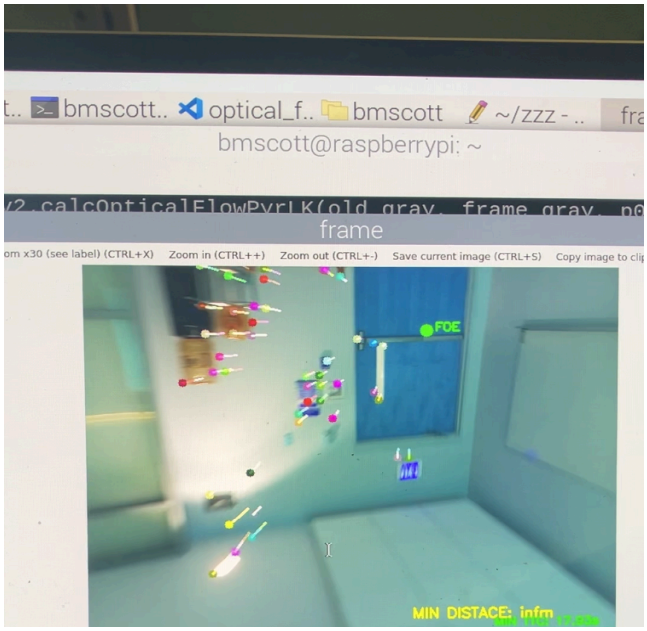
We also encountered many challenges in our optical flow CV model. One large challenge was collision detection responsiveness vs resilience to false positives. Since we had our distance to collision set to a threshold value, and if we dip below that threshold, we actuate the brake, false positive detection would cause a false braking signal. We utilized this setup, since a more delayed metric of multiple frames led to latency that was too large for successful braking, however this lack of resilience to brief outliers of short distance was a persistent problem. Our next challenge was pertaining to feature detection of the model, especially in visually busy environments. Since Lucas-Kanade uses contrast to determine features, we found that in many environments, it would detect many features on distant objects such as trees or buildings, and thus have an extremely high rate of false positives, thus actuating the brake. Finally, we observed very low reliability at lower speeds, since with low forward movement, jitter and tilt of the bike would contribute heavily to the movement of features in the image, leading to again the issue of too many false positives.

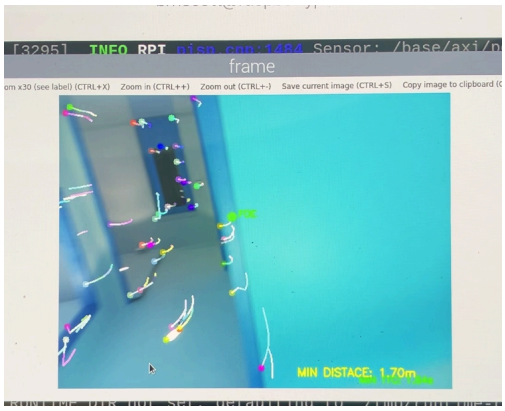
8. Experimental section

It is important to note that our experiments were performed at lower speed within a controlled indoor environment. This is because in an outside environment, the Lucas-Kanade method of optical flow picked up too many high contrast features around the corners and edges of the view, so the false positive rate (false brake actuation) was too high to obtain reasonable results about braking distance or accuracy in

response to real collisions. Additionally, getting an exact same speed between trials on the bicycle was difficult, so we used many repeated trials, until we got speeds within a reasonable margin for trials.

2.





Trial Results:

A	B	C	D
	Low speed (meters)	Med speed	High Speed
1	0.5	3.3528	1.8288
2	0.35	1.2192	0.9144
3	2.1336	1.524	1.2192
4	1.2192	1.8288	3.9624
5	0.762	4.2672	0
	0.3 m/s	0.5 m/s	0.6 m/s

9.

The above tables demonstrate the results of our trials done indoors with the above experimental setup. Due to a high rate of false positives causing braking before we could get up to trial speeds, it was

unfeasible to do a large amount of trials. The above diagrams display the distance to collision in meters, for 5 trials of traveling at the above indoor experimental setup, with distances above 3m being flagged in red, since our detection threshold was 3m, so it is unclear if our detected feature was the desired obstacle. Our one reading of 0 corresponds to a collision with the wall. The m/s figure below for each trial is accurate to +-0.025 m/s.

It is important to note that the above tables consist of the limited amount of quantitative data we were able to obtain, considering that our false positive rate for braking was so high it was untenable to run any longer trials.

From our results, we can conclude that within a controlled environment, it is possible to automatically actuate bicycle brakes off of a single camera computer vision model, with the limited compute power of the raspberry pi. We also conclude that to obtain a robust CV actuated braking system to work in an outdoor environment, more compute power to apply more computationally expensive computer vision, and/or more and varied types of sensors such as lidar may be required.

1. What did you accomplish? How does this demonstrate key concepts in this class or further the development of embedded systems?

- We demonstrate interrupt based detection handling for our collision sensor, as well as utilization of a closed control loop for the braking system, since the wheel speed is directly affected by the braking motor, and the braking motor is controlled by the wheel speed during braking events. The process variable is the wheel speed, the software is the controller, the motor is the actuator, and the collision sensor the sensor.
- We additionally demonstrate another potential application of computer vision in embedded systems, and extend the work done on collision detection via computer vision in embedded settings.
- Finally, we demonstrate the use of wood in prototyping for actuated systems, and mounting of components when there is not access to more robust machining environments.

2. Discuss any unfinished parts of the implementations and/or next steps if you were to continue working on the project.

- Next steps for braking include utilization of a different braking mechanism, ideally hydraulic disk brakes, with a hydraulic control system, in order to obtain more braking force and fluctuation, by utilizing this with a wheel speed encoder, we may be able to attain more fine grained measurements of wheel velocity and control to implement ABS. Additionally, re-implementation of a human controlled rear brake with a pass through feature would be ideal.
- For our computer vision, we would ideally first implement ransac image stabilization, thus hopefully reducing the effect of natural movement of the bicycle due to the rider or the terrain, and thus reducing FPR of the model. After this implementation, we would be able to sample more rapidly for motion detection, since jitter in the image would not contribute to features in Lucas-Kanade, from the above combined metrics, we would be able to fine tune the model further, and potentially change to a consecutive frames model instead of a single detection with rolling average, thus reducing our false positive rate further.