I-MOVE: Independent Moving Objects for Velocity Estimation

Jonathan Schwan Akshay Raj Dhamija Terrance E. Boult
University of Colorado, Colorado Springs
{jschwan2 | adhamija | tboult} @ vast.uccs.edu

Abstract

We introduce I-MOVE, the first publicly available RGB-D/stereo dataset for estimating velocities of independently moving objects. Velocity estimation given RGB-D data is an unsolved problem. The I-MOVE dataset provides an opportunity for generalizable velocity estimation models to be created and have their performance be accurately and fairly measured. The dataset features various outdoor and indoor scenes of single and multiple moving objects. Compared to other datasets, I-MOVE is unique because the RGB-D data and speed for each object are supplied for a variety of different settings/environments, objects, and motions. The dataset includes training and test sequences captured from four different depth camera views and three 4K-stereo setups. The data are also time-synchronized with three Doppler radars to provide the magnitude of velocity ground truth. The I-MOVE dataset includes complex scenes from moving pedestrians via walking and biking to multiple rolling objects, all captured with the seven cameras, providing over 500 Depth/Stereo videos. To access the dataset please visit www.vast.uccs.edu/imove

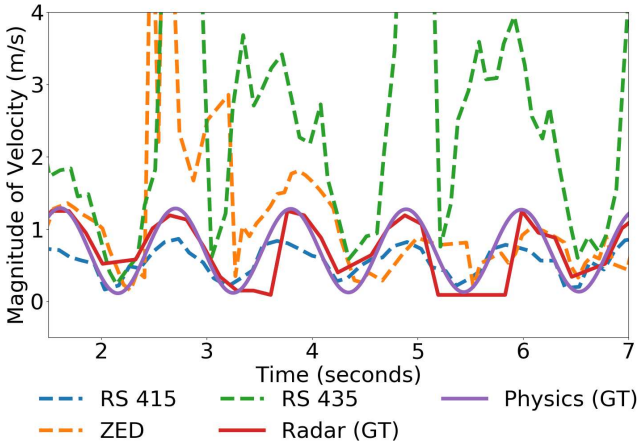


Figure 1: I-MOVE PROBLEM *While there has been considerable research in velocity estimation, this paper, with the help of the proposed I-MOVE dataset, highlights the need for improvement in the field. In the above plot, we consider a swinging pendulum and record its instantaneous velocity with the radar. We use the radars and physics laws to estimate the ground truths. As evident from the plot, standard approaches for velocity estimations using vision-based systems do not provide values remotely close to either of the ground truth estimates. Further details are available in Sec. 5*

1. Introduction

In recent years, the field of video-based computer vision has lead to exploration of some interesting problems, such as tracking [3, 35, 23, 25], localization and mapping [41, 4], action recognition, as well as sentiment analysis [6, 32]. We present the relatively unexplored task of motion parameter estimation. Even though motion parameters are useful for a large variety of applications, estimating them from videos has not been studied extensively. Motion parameters are a necessary component in numerous applications, such as robotic navigation, [10, 1] collision detection, [3, 18, 20], drone visual tracking[43, 38], and car velocity estimation for speed monitoring or accident prevention [16, 19, 5, 42]. In problems, such as collision detection, it is not only necessary to take into account velocity, but also that of other surrounding independently moving objects. Similarly, in robotics, if one wishes to enable a robot to interact with an independently moving object (e.g. flying ball, frisbee, or drone), the motion parameters of these objects need to be accurately estimated to understand the trajectory and predict the future location of the objects successfully. Another important application area for motion parameter estimation is in the realm of sports. Numerous sports performance analysis of athletes relies on velocity and acceleration information. Most obviously, sports where speed is the main component (running, biking, swimming, etc.). However, this is also a key component for sports, such as weightlifting, where the athletes are looking for their lift force. Requiring acceleration information to estimate and find the best posture/lift technique. Similarly, motion parameters may also be useful for training purposes in various sports, such as skiing, snowboarding, or skateboarding. Having related data for these problems

has become so crucial that many synthetic environments have been created [17, 40, 14]. These synthetic environments have greatly helped people approach the problem, but in the unconstrained real-world environments, these tasks are much more complicated than the research environments. While most of the constrained application areas could either document motion parameter information using specialized sensors, such as Inertial Measurement Unit (IMUs) or from egocentric videos, these are not viable for unconstrained scenarios because both of these approaches need to have access to the object in motion. With a task, such as estimating the instantaneous velocity of a flying ball, neither IMUs nor egocentric videos may be used [8, 7, 13, 27]. Since data from cameras can be easily accessible for such problems, they become the logical choice to create a more useful and robust method for motion parameter estimation.

Currently, there are no existing, publicly available datasets that provide velocity estimation for such complicated tasks; for this reason, we are introducing a new dataset, I-MOVE. **Compared to the existing datasets, I-MOVE is unique in that it provides all of the following:**

(a) First to provide instantaneous velocity ground truth

(b) Contains a variety of objects and motions

(c) Includes both indoor and outdoor scenes

(d) Captured using many types of cameras

(e) Simultaneous captures from a variety of viewing angles

(f) Validation procedure for velocity ground truth

2. Related Datasets

Non-RGB-D vision-based velocity estimation has been studied for decades [21, 31, 29]. In recent years, with application of computer vision algorithms to the domains of robotics [39, 22], autonomous driving cars [15, 24], vehicle velocity estimation/monitoring[16, 19, 5, 42], and drones [10], the number of works attempting to estimate motion parameters has grown dramatically [11, 33]. As a result, the need for these datasets has also grown greatly [42, 37, 26, 9, 36]. Many of these works differ in the primary purpose of the dataset, the ground truth supplied by the dataset, the environments in which the datasets take place, and the objects for which motion parameters are estimated. In this section, we first recognize the datasets that either focus on motion parameter estimation or on a related task. These datasets generally use RGB data or data acquired from non-vision systems. We then discuss the most similar datasets within the RGB-D realm.

2.1. RGB or Motion Parameter Only Datasets

In the purely RGB world, there are existing datasets, such as the Multi-Object Tracking (MOT) Benchmark dataset [30]. This dataset has videos of high pedestrian traffic areas. MOT is an excellent dataset for testing tracking in complex environments. An effective tracking algorithm for this dataset, which would allow you to estimate the pixel space velocity. However, if one wished to estimate the velocity in a meaningful metric (i.e. meters per second) it would not be possible to do so accurately without substantial intrinsic and extrinsic calibration.

The Human Activity Recognition dataset [2] contains smartphone accelerometer information collected by numerous people performing various tasks, such as sitting, walking, and going upstairs. This dataset contains no images/video and was created with the intent of generating a model that could predict activity solely from the accelerometer information. Because the problem presented in this paper aims to estimate the motion parameters of an object from a video, this dataset cannot be utilized.

An additional human activity recognition-based dataset is the UTD-MHAD [9], which contains both IMU and video information. This dataset contains 27 actions performed by 8 subjects (4 females and 4 males). Each subject repeated each action four times. The actions include motions, such as a knock on a door, sit to stand, and stand to sit. Because these actions are extremely limited in their variety and the length of time they are performed, they are not as desirable to estimate motion parameters. The dataset also contains no ground truth for velocity or motion parameters other than that provided via the IMU.

Another interesting dataset is the HumanEva dataset [36], which is a synchronized video and motion capture dataset. It consists of four subjects performing a set of six predefined actions three times (twice with video and motion capture, and once with motion capture alone). This dataset was intended to be used to improve existing three-dimensional pose estimation and action recognition. If one wished to use the dataset for motion parameter estimation, it would be challenging due to the lack of velocity ground truth, variety of speeds, and types of motions/settings for the data.

2.2. RGB-D Datasets

While there are a variety of RGB-D datasets, to the best of our knowledge, there is no RGB-D dataset that contains an adequate velocity ground truth to evaluate the performance of an algorithm/method accurately. The most similar dataset is the one proposed in the paper, A Benchmark for the Evaluation of RGB-D SLAM Systems [37]. The dataset contains the color and depth images from a Microsoft Kinect sensor along with the ground truth trajectory of the sensor. The ground truth trajectory was obtained from a high-accuracy motion capture system with eight high speed tracking cameras plus the accelerometer data from the Kinect. However, since the Kinect has limited perfor-

Figure 2: DATASET COLLECTION SETUP *The I-MOVE dataset is aimed at vision-based estimation of the velocity of moving objects. Above is the setup used to record the RGB-D data of a moving pendulum along with its instantaneous velocity. To record the vision-based information, we use a variety of stereo camera devices. Each of the cameras, even when belonging to the same family of depth calculation technique, differ in specifications as summarized in Table 1 and Sec. 3.1. For recording the velocity ground truths, we use three Doppler radars, specifically three OmniPreSense radars that are positioned such that they are perpendicular to the motion of the object. Certain scenes, such as pendulum, dropped object, and rolling object on ramp, incorporate additional ground truth from physics-based calculations further detailed in Sec. 3.*

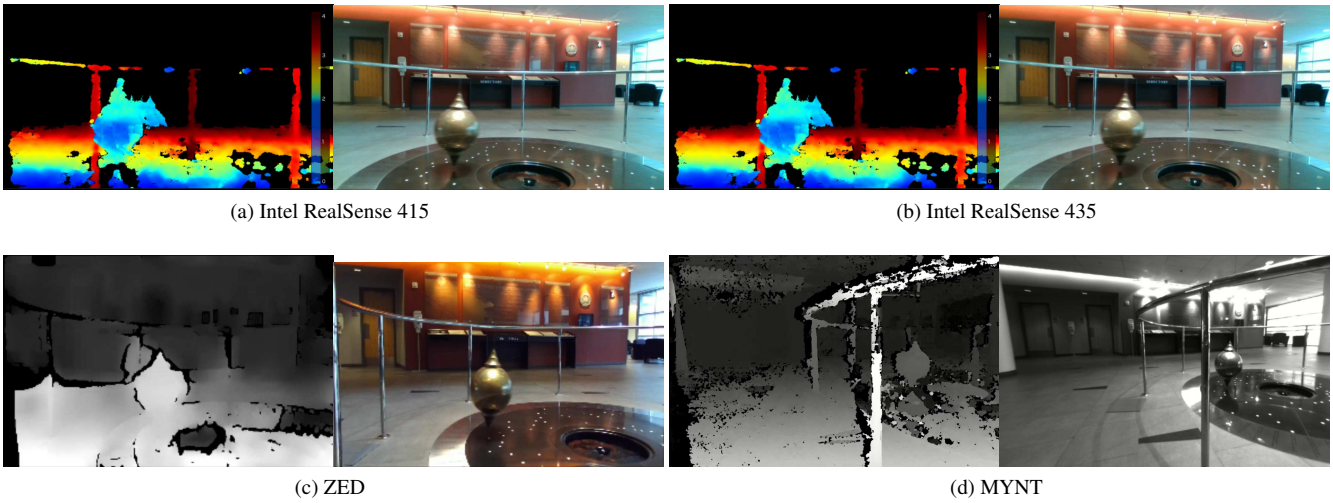
Camera	Resolution	FPS	Horizontal-FOV	Vertical-FOV	Type
ZED	2208 x 1242	15	90°	60°	Stereo RGB
Stereo GoPro (Standard)	4096 x 3072	15	122.6°	94.4°	
Stereo GoPro (Modified)	4096 x 3072	15	54.1°	32.1°	
Stereo GoPro (Modified)	4096 x 3072	15	64.7°	39.3°	
RealSense 415	1280 x 720	30	85°	58°	Stereo RGB-D (IR)
RealSense 435	1280 x 720	30	63°	42.5°	
MYNT	1280 x 720	60	122°	76°	Monochromatic IR

Table 1: CAMERA INFORMATION *Each camera used in the data collection process is different to ensure the velocity estimation model generated is more likely to generalize to different devices, perspectives, and scenes. The cameras all differ in field of view (FOV), in addition, there are three different resolutions, frames per second, and types of depth calculation methods. This variety also helps provide an accurate comparison between each device for settings, objects, and motions with which they perform best.*

mance in outdoor environments, the dataset was restricted to indoor use only. Moreover, the dataset only contained a single type of object. Hence, even if someone would attempt to create a system for motion parameter estimation on this dataset, it may not generalize well on other objects. The DIML RGB-D dataset [26] also contains data collected with a Kinect. However, this database does include indoor and outdoor video in addition to object segmentation, making it more plausible to conduct tests for motion parameter estimation purposes. But this dataset does not contain any velocity ground truth. The dataset also only contains single camera views, making it likely that any system created to estimate motion parameters on this dataset may not translate well to data from a different camera source.

3. The I-MOVE Dataset

I-MOVE, unlike any other dataset, contains various objects subjected to a variety of motions under varying scenes, lighting, and recorded with seven different cameras. The seven cameras may be divided into three different categories, stereo RGB, RGB-D & stereo, and monochromatic with IR depth. The variety of cameras capturing the same scene simultaneously allows the user to compare the performance not just of various algorithms but also of various cameras, allowing one to decide on the best camera for a specific application. The variety in objects and scenes also ensures each video differs widely not only in shape and size of the object but also motion paths, making them suitable for training with deep learning-based techniques where varying data is useful. A break down of what the current I-MOVE dataset contains can be found in Table 2. I-MOVE



(a) Intel RealSense 415 (b) Intel RealSense 435

(c) ZED (d) MYNT

Figure 3: SAMPLE IMAGES FROM OUR DEPTH CAMERAS *Though all the above images provide us depth information from the scene, each of the cameras differ in specifications such as field of view and frames/sec. These variations can be used in the future to answer more daunting application-oriented questions, such as what camera should be used for a specific problem regarding a small object, fast-moving object, or indoor versus outdoor.*

also includes the instantaneous velocity of the object in all of the above scenarios. An essential aspect of any given dataset is the accuracy of its annotations. Since I-MOVE uses specific sensors for providing instantaneous velocity, it becomes necessary to validate the performance of these sensors. In order to understand the extent of any errors due to these sensors, the dataset includes some carefully crafted experiments that can be closely related to the laws of physics. As demonstrated in later sections, these experiments are used to validate the performance of the velocity sensors.

3.1. Cameras & Calibration

In order to record the same object movement from various viewing angles, we used seven different cameras that can be divided into the following three categories. *a*) **Stereo RGB:** This category consists of a ZED camera and three stereo GoPro Hero 3s (six GoPros in total because there are two GoPros per stereo setup). From the three GoPro stereo cameras, we modify the configuration of two to provide us significant variation in the horizontal field of view. *b*) **Stereo RGB-D (IR):** For the IR-based RGB-Ds with stereo, we used Intel RealSense 415 and Intel RealSense 435 cameras. *c*) **Monochromatic IR:** For the monochromatic camera with IR depth, we used the MYNT Eye S. Further details on the cameras may be found in Table 1. For calibrating all of the above cameras, the intrinsic and extrinsic calibration information was collected using the checkerboard approach commonly performed with OpenCV [34, 12]. These calibration video segments are included as part of the dataset in case researchers wish to utilize them. One such calibration segment can be found for each significant change in lighting, background, or experimental setup. For some of the cameras, such as the ZED and Intel RealSense, calibration options are also available within the SDKs; these can be used if a single world coordinate space is not necessary.

This variety in cameras enables researchers to explore the best camera according to the scenario they are attempting to address. The variety of scenarios may include the following: object size, object speed, environmental conditions (indoor/outdoor).

Small objects: For the purpose of velocity estimation, it is necessary for a system to be able to track the moving object accurately across frames. If an object is smaller, it may result in increased difficulty for tracking the object. With cameras that have a lower resolution, this inconsistent tracking would also significantly decrease the accuracy of the depth estimation for the object. In such a scenario, a high-resolution camera like the GoPro with a 4K resolution may provide much higher performance than the lower resolution RGB-D cameras. Stereo cameras with a greater distance/baseline between each lens can also improve the depth estimation accuracy for objects that are smaller or further away.

Fast-moving objects: Similarly, tracking fast-moving objects can also be extremely difficult. If their size in the

	Scene	Number Objects/ Differences	Num. Depth Videos
Indoor	Intrinsic Calibration	2	14
	Extrinsic Calibration	2	14
	Rolling Object Single Ramp	15	105
	Rolling Object Two Ramps	5	35
	Rolling Object Two Ramps (facing ramp)	4	28
	Biking	2	14
	Variety movements (Walking, Skateboarding, Biking)	2	14
	Pedestrians / Walking	3	21
	Skateboarding	2	14
	Object Throwing	3	21
	Pendulum	15	105
	Object Drop	3	21
Outdoor	Intrinsic Calibration	1	7
	Extrinsic Calibration	1	7
	Pedestrian	3	21
	Object Drop	2	14
	Rolling Object Single Ramp	9	63
	Rolling Object Two Ramps	10	70

Table 2: I-MOVE DATASET BREAKDOWN *The I-MOVE dataset is presented above by scene type, then by the respective amount of differences both in object/setup (i.e. changes in ramp height or object swung on pendulum). The final column is the corresponding quantity of depth videos associated with each type of scene (there are seven times the number of each scene). If individual RGB videos are desired, there are 14 times the number of each corresponding scene. The I-MOVE dataset contains scenes varying in objects, motions, settings (indoor and outdoor), and complexity (from a single ball rolling down a ramp to a biker, pedestrian, and skateboarder all in the same scene). This large variety and quantity in the dataset allow for models to be generated/created that are more likely to generalize to other scenes and applications as well as testing the velocity estimation model on both the simple and more complex ends of the spectrum.*

frame is large enough, this may not be as significant of an issue, but for most practical cases, the object will be moving so quickly that there is a significant amount of blur, making it difficult to track and lowering the quality of the depth estimation. Higher frame-rate cameras, such as the MYNT and Intel RealSenses, should be less affected by this problem. For lower frame-rate devices, this problem can be much more significant depending on the accuracy of the object tracker and the form of depth estimation used.

Indoor vs. Outdoor / varying illumination: In many practical applications of velocity estimation, there is frequently fluctuation in lighting. Where indoor scenarios often help improve IR-based depth devices accuracy, they also increase the amount of blur likely for moving objects. In outdoor situations, the light may be so bright that it lowers the depth accuracy of IR-based depth devices by a significant margin. However, it can potentially improve the accuracy of stereo-based devices such as the ZED. Outdoor situations also increase the likelihood of lighting changes during a scene/period of time where velocity estimation would be needed, increasing the likelihood that object trackers would have greater difficulty in maintaining a precise bounding box for the object(s) of interest.

3.2. Collecting Ground Truth Velocity

Since the I-MOVE dataset is aimed at providing standardization for prediction of velocity for a moving object, it also needs to provide instantaneous velocity values. To achieve the instantaneous velocity, we use three doppler radars. Specifically, we use OmniPreSense doppler radars, which provide the magnitude of the velocity of objects within their 78° wide beams. Each radar provides nine magnitudes of velocity sorted by the return radar strength.

To ensure the accuracy of the values provided by the radars used in our experiments, we design specific experiments where velocity can be estimated using the laws of physics. These experimental setups include dropping an object, rolling an object down a ramp, and swinging an object like a pendulum. With these known physics-based setups, it becomes possible to use physics equations to find the instantaneous velocities for each setup, which are used to estimate the error in the radars. The equations used for each of the setups required the more complex instantaneous velocity calculations to be used as opposed to the more common final velocity equations. This is because we wanted to obtain velocity for each frame/image within the videos collected.

Free fall (drop): The velocity data for the object drop was computed using the commonly known equation $V = \frac{1}{2}gt^2$, where g is the gravitational acceleration ($9.8m/sec^2$) and t is the time since the object was released. Because the radars and cameras are returning timestamp information, the velocity is easily calculated and compared at any given timestamp.

Rolling down a ramp: Calculating the instantaneous velocity of a rolling ball/object is slightly more complicated. Aspects, such as friction between the ball and the ramp, can prove to be very inaccurate and tedious to measure. Since the goal of using these physics-based experiments is not to calibrate the radar but to get an understanding of the validity of the radar estimates, we assume the ramp and air to provide zero friction/resistance concerning the ball. The

final velocity (velocity when the rolling object reaches the end of the ramp) is calculated using the following equation $V_{final} = \sqrt{\frac{10}{7}gh}$, where g is once again gravity and h is the height of the ramp. Now that the final velocity is obtained, and given that we know the initial velocity as zero, we can find the average acceleration using the equation, $a = \frac{V_{final}}{\Delta t}$ by dividing the final velocity by the change in time (the time it takes to reach the bottom of ramp). We can then use this average acceleration to find the velocity at any point between the object starting down the ramp and reaching the ground. To do this, we use the following equation $V_t = at$. This equation multiplies the average acceleration down the ramp a by the time since the release of the object (t), to provide us the instantaneous velocity of the object while rolling down a ramp.

Swinging object / pendulum: To calculate the instantaneous velocity for a moving object, we needed to measure the length (L) of the string to which the object was attached. The time taken by an object to complete a swing period can be determined using the equation: $P = 2\pi\sqrt{\frac{L}{g}}$. We can then determine the angle of the object with the vertical θ at any given time t as $\theta_t = \theta_{Max} cos(\frac{2\pi}{P}t)$, where θ_{Max} is the highest point of the swing (or the initial drop angle), P is the period of the swing, and t is the time since the initial drop, at which we are trying to estimate the instantaneous velocity. Given this angle θ_t, we can then find the instantaneous velocity of the pendulum with the equation: $V_t = \sqrt{2.g.L(\cos\theta_t - \cos\theta_{Max})}$. V_t is the instantaneous velocity at time t.

Now that we are able to solve for the instantaneous velocity of the pendulum, we apply this to each time-step recorded pendulum data. The accuracy of this velocity data is also significantly improved by the fact that we applied this to pendulum drops of 20°or less, making it a simple small-amplitude pendulum problem and improving the data generated via the prior equations. If the release had been at a greater angle, it would allow more room for free fall, air resistance, and rotation/circular swing to affect the accuracy of the equations used.

Multiple moving objects: For objects which have various independently moving parts, such as arms or legs of a person, collecting ground truth velocity becomes nontrivial, especially without specialized systems, such as MOCAP.

Each of our radars provided nine velocity estimates, sorted by the magnitude returned. Consolidating this data is tedious and hence has been left to future work. For the scope of this work, we use the velocity reading for the most significant radar returns, which is roughly the largest nearby objects.

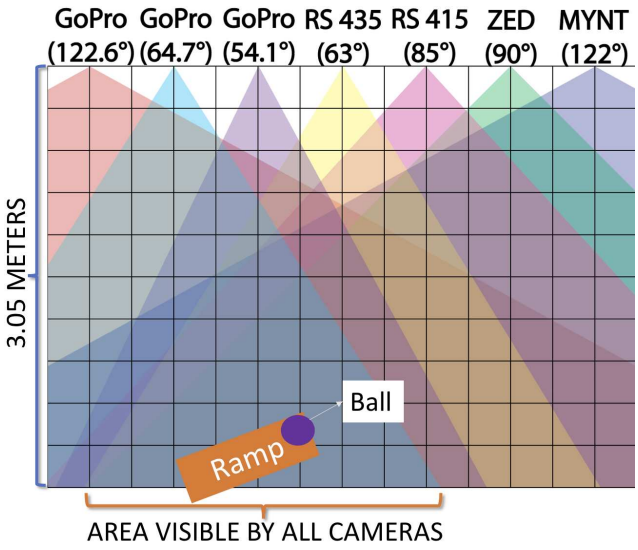


Figure 4: SKEMATIC FOR HORIZONTAL FIELD OF VIEW OF EACH CAMERA *Each box represents a 1 foot x 1 foot square and, as can be seen in the figure, the cameras are placed two feet apart horizontally. The figure also shows that object has to be slightly less than five feet away from the center camera (Intel RealSense 435) in order to be in the field of view of all the cameras. We conducted various scenes with object's motions (such as a ball rolling down a ramp) within the field of view of all devices as can be seen above.*

3.3. Setup

The apparatus used for data collection was meticulously crafted to provide reliable results in various locations/scenes. The seven cameras were mounted on an angle bar, as can be seen in Figure 2. This allows us provides a significant difference in the X-axis and a slight difference in the Z-axis for each camera.

The cameras were mounted identically at each location such that each camera's abilities can also be evaluated (range, quality, accuracy, etc.) in different settings.

Due to the wide variety in object size, scene layouts, and environments where data was collected, the apparatus was created to accommodate these differences. The cameras all have a different field of view, making the order and spacing of them vital components, so the key actions/movements of the object can be captured by every device.

Given our purposeful placement based on the field of view of the cameras, see Figure 4, we were able to obtain a two foot spacing between each camera. This allowed for a fairly significant difference in camera perspective while still having all cameras capturing the object, and its most valuable movements. The set up was arranged from left to right (facing the lens) as follows: Stereo GoPro (standard), Stereo GoPro (modified 64), Stereo GoPro (modified 54), Intel RealSense 415, Intel RealSense 435, ZED,

MYNT Eye S (Monochromatic), which can be seen in Figure 2. This rig allows us to have a portable and consistent system, which is essential given the numerous locations/environments where the data was collected.

3.4. Synchronization of Sensors

In addition to calibration, synchronization was also essential because the same motion ground truth was used for different cameras and perspective. For this reason, it was also crucial to ensure the radars providing the ground (that is not obtainable via physics-based setups/equations) were time synced with the cameras so that the ground truth was accurately applied to the appropriate frame from each stereo camera setup. To do this, we time synced the computers used to collect the information from both OmniPreSense radars also to collect the Intel RealSense 435 and 415 data. Both radars were set to return their timestamp information in addition to the velocity magnitudes. This synchronization allowed us to use a flash event, where we utilized a camera flash lasting four milliseconds. The flash allowed the moment to be visually captured by all the stereo cameras thus using the frame(s) with flash to appropriately sync the velocity information from the radars to the corresponding camera frames.

4. Evaluation

Our framework is a platform for a fair comparison of state-of-the-art velocity estimation methods. We provide authors with standardized ground truth data, evaluation metrics, and scripts to ensure that the velocity estimation method is isolated from other components. This approach towards evaluation provides a more fair and accurate comparison. We employ the commonly known yet valuable Mean Absolute Difference (MAD) comparison metric.

$$MAD = \frac{1}{n}\sum_{i=1}^{n}|x_{i1} - x_{i2}|$$

Where x_{i1} is the estimated magnitude of the velocity and x_{i2} is ground truth magnitude of velocity, with n being the number of ground truth measurements for each scene. The evaluation script uses the ground truth radar data associated with the scene/video and subtracts the corresponding velocity estimation given for each timestamp. For frames in which there is not a velocity estimation, the script will interpolate based on the existing data. For each video, the MAD will be initiated/calculated from the beginning of the object's motion until either a specified amount of time has passed or the motion of the object has ceased. Because of the large number of scenes and variety in motion and object size, we use the ramp, drop, and pendulum scenes to adequately evaluate each velocity estimation method. The drop and ramp scenes have different yet still mostly linear motions, whereas the pendulum scene provides more complex motion and changes in the direction of the motion. Using these scenes paired with an outdoor example to test the effectiveness of the model in a different setting allows us to have a better performance measure of each velocity estimation model tested on the dataset.

5. Baseline

As a starting point for the I-MOVE dataset, we used a simple multi-stage approach to estimate the velocity vector and its magnitude. We first automatically detect the moving object's location, then implement a tracker to return the position of the object throughout the scene. Finally, we implement a Kalman filter to estimate the speed of the object for each frame. We briefly outline each component and the results of our baseline, but for greater detail and access to the code, please visit the VAST GitHub.

5.1. Object Detection

In order to identify the object automatically, we utilize a motion detection approach to identify the object(s) of interest and initialize the tracker with their location. We first take the RGB / color frame and apply a mild Gaussian blur in order to make smaller, insignificant motions from each frame unnoticed. We then apply the built-in OpenCV Background Subtractor KNN and save the first few frames. We use these frames to compare against future frames for the difference in pixels. When there is a significant difference/movement from the initial frames, we identify the area/bounding box around the movement and initialize the tracker with the bounding box corners. This approach, although effective, does have the limitation of missing some of the initial movement. For shorter object tracking events/motions this proved to be a significant issue.

5.2. Object Tracking

Once the bounding box in returned by the motion detection algorithm we use it as the initializing bounding box for the Discriminative Correlation Filter Tracker with Channel and Spatial Reliability (CSRT) [28]. The CSRT tracker tracks the object throughout the RGB frame, returning its bounding box location in two-dimensional space. We then take the center pixels of this bounding box and, using the corresponding camera's API, project the two-dimensional pixel to its three-dimensional coordinates. The three-dimensional coordinates are then averaged and returned as one three-dimensional coordinate for the object in the respective frame. These three-dimensional coordinates, along with their frame's timestamps are then used as inputs to a Kalman filter.

Source	Pendulum		°Ramp		Drop		Outdoor 8° Ramp	
	Radar (R)	Physics (P)	R	P	R	P	R	P
Radar	-	0.152	-	0.016	-	0.068	-	0.023
ZED	1.090	1.108	0.108	0.274	1.846	1.215	0.255	0.245
RealSense 435	1.743	1.589	0.441	0.393	1.868	1.216	0.506	0.410
RealSense 415	0.283	0.275	0.199	0.153	2.083	2.145	0.269	0.208

Table 3: BASELINE RESULTS *For each physics-based setup we evaluated a scene using the Mean Absolute Difference (MAD) with meters per second as the units. To evaluate the performance of the radar, we compared its velocity estimates to those from the physics-based equation. Each of the above cameras (ZED, Realsense 415, and 435) were evaluated against both the radar data and the physics-based velocity estimates. The results make it clear that there are still significant improvements to be made before the problem is solved. There are somewhat reasonable results for each of the cameras in the indoor and outdoor ramp scenes; however for the remaining scenes (pendulum and dropped object), especially for the high speed of the dropped object, results are substantially worse.*

5.3. Kalman Filter

Speed estimation of the object being tracked is done with a Kalman filter. Our Kalman filter assumes constant acceleration and with a state vector of:

$$X = \begin{bmatrix} x & y & z & \dot{x} & \dot{y} & \dot{z} & \ddot{x} & \ddot{y} & \ddot{z} \end{bmatrix}^T$$

Within the Kalman Filter, we used the state transition matrix $A =$

$$\begin{bmatrix} 1 & 0 & 0 & \Delta t & 0 & 0 & \frac{1}{2}(\Delta t)^2 & 0 & 0 \\ 0 & 1 & 0 & 0 & \Delta t & 0 & 0 & \frac{1}{2}(\Delta t)^2 & 0 \\ 0 & 0 & 1 & 0 & 0 & \Delta t & 0 & 0 & \frac{1}{2}(\Delta t)^2 \\ 0 & 0 & 0 & 1 & 0 & 0 & \Delta t & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & \Delta t & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & \Delta t \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

where change in time (Δt) used in the state transition matrix is calculated by finding the difference in the time between each frame's timestamps. The Kalman is updated using the 3D point measurement (estimated location) from the information returned by the CSRT tracker. The magnitude of the velocity of the object is then computed from the Kalman estimated velocity for each frame, by taking $\sqrt{\dot{x}^2 + \dot{y}^2 + \dot{z}^2}$, where $\dot{x}$, $\dot{y}$, and $\dot{z}$ are from the state vector X. This approach allows us to estimate the velocity for every frame/at every timestamp provided by the corresponding camera.

5.4. Current Results

Though tests were conducted on numerous videos, we show a few examples from our current baseline in Table 3 and Figure 1. For easier and more gradual situations, such as a rolling object going down a slightly inclined ramp, viable speed estimations can be made. However, for more complex and quick motions, there is still a large and unsolved problem. For example, although we were able to obtain fairly good estimations for an approximately $\approx 8^\circ$ ramp in the indoor and outdoor videos with the RealSense 415, that sensor was completely unable to return good estimations for a dropped/falling object. We saw similar difficulties with dropped/falling objects in the other cameras as well. The RealSense 435 and ZED cameras also seemed to struggle with the pendulum videos where the greatly alternating depth caused problems when the tracker's bounding box was slightly off of the pendulum. These baseline results for rather simple motions make it clear there is much work to be done on multiple aspects of this problem, from the automatic object tracker to the motion estimation algorithm/approach itself. For multiple moving objects (not shown), the baseline results are even worse.

6. Conclusion & Future Work

This paper presented a novel, challenging set of motion sequences and their corresponding magnitude of velocities within the I-MOVE dataset. This dataset is intended to help researchers progress and refine their approaches to produce more robust velocity of estimation of an object being tracked. We identify several drawbacks and limitations within the existing datasets in addition to explaining the differences between our dataset and ground truths. We also explain how no pre-existing datasets contain all of the necessary information to adequately approach the problem of independently moving object velocity estimation. The problem of velocity estimation for an independently moving object given RGB-D data is certainly not currently solved. However, given public access to the I-MOVE dataset it is now much more feasible for people to create and test their approach to solving this problem. In future works we plan to use the three radars placed on a calibration board to develop novel stereo-radar approaches to extract three dimensional velocity vectors as an additional ground truth. We also intend to use moving cameras so the dataset will have the additional diversity of both stationary and dynamic camera data. All updates and modifications to the dataset will be available at www.vast.uccs.edu/imove

References

[1] Vision-only egomotion estimation in 6dof using a sky compass | Robotica | Cambridge Core.

[2] D. Anguita, A. Ghio, L. Oneto, X. Parra, and J. L. Reyes-Ortiz. A Public Domain Dataset for Human Activity Recognition using Smartphones. In *ESANN*, 2013.

[3] S. Atev, H. Arumugam, O. Masoud, R. Janardan, and N. P. Papanikolopoulos. A vision-based approach to collision prediction at traffic intersections. *IEEE Transactions on Intelligent Transportation Systems*, 6(4):416–423, Dec. 2005.

[4] E. Brosh, M. Friedmann, I. Kadar, L. Yitzhak Lavy, E. Levi, S. Rippa, Y. Lempert, B. Fernandez-Ruiz, R. Herzig, and T. Darrell. Accurate Visual Localization for Automotive Applications. pages 0–0, 2019.

[5] H. Caesar, V. Bankiti, A. H. Lang, S. Vora, V. E. Liong, Q. Xu, A. Krishnan, Y. Pan, G. Baldan, and O. Beijbom. nuScenes: A multimodal dataset for autonomous driving. *arXiv:1903.11027 [cs, stat]*, Mar. 2019. arXiv: 1903.11027.

[6] C.-Y. Chang, D.-A. Huang, Y. Sui, L. Fei-Fei, and J. C. Niebles. D3tw: Discriminative Differentiable Dynamic Time Warping for Weakly Supervised Action Alignment and Segmentation. pages 3546–3555, 2019.

[7] A. Chatzitofis, N. Vretos, D. Zarpalas, and P. Daras. Three-dimensional monitoring of weightlifting for computer assisted training. In *Proceedings of the virtual reality international conference: Laval virtual*, page 3. ACM, 2013.

[8] A. Chatzitofis, D. Zarpalas, and P. Daras. A computerized system for real-time exercise performance monitoring and e-coaching using motion capture data. In *Precision Medicine Powered by pHealth and Connected Health*, pages 243–247. Springer, 2018.

[9] C. Chen, R. Jafari, and N. Kehtarnavaz. UTD-MHAD: A multimodal dataset for human action recognition utilizing a depth camera and a wearable inertial sensor. In *2015 IEEE International Conference on Image Processing (ICIP)*, pages 168–172, Sept. 2015.

[10] H.-M. Chuang, T. Wojtara, N. Bergstrm, and A. Namiki. Velocity Estimation for UAVs by Using High-Speed Vision. *Journal of Robotics and Mechatronics*, 30(3):363–372, 2018.

[11] H. Coskun, F. Achilles, R. DiPietro, N. Navab, and F. Tombari. Long Short-Term Memory Kalman Filters: Recurrent Neural Estimators for Pose Regularization. pages 5524–5532, 2017.

[12] A. Datta, J.-S. Kim, and T. Kanade. Accurate camera calibration using iterative refinement of control points. In *2009 IEEE 12th International Conference on Computer Vision Workshops, ICCV Workshops*, pages 1201–1208. IEEE, 2009.

[13] M. Einfalt, D. Zecha, and R. Lienhart. Activity-conditioned continuous human pose estimation for performance analysis of athletes using the example of swimming. In *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 446–455. IEEE, 2018.

[14] L. Fan, Y. Zhu, J. Zhu, Z. Liu, O. Zeng, A. Gupta, J. Creus-Costa, S. Savarese, and L. Fei-Fei. SURREAL: Open-Source Reinforcement Learning Framework and Robot Manipulation Benchmark. In *Conference on Robot Learning*, pages 767–782, Oct. 2018.

[15] Fangjun Jiang and Zhiqiang Gao. An adaptive nonlinear filter approach to the vehicle velocity estimation for ABS. In *Proceedings of the 2000. IEEE International Conference on Control Applications. Conference Proceedings (Cat. No.00CH37162)*, pages 490–495, Sept. 2000.

[16] J. Farrelly and P. Wellstead. Estimation of vehicle lateral velocity. In *Proceeding of the 1996 IEEE International Conference on Control Applications IEEE International Conference on Control Applications held together with IEEE International Symposium on Intelligent Contro*, pages 552–557, Sept. 1996.

[17] F. Fei, Z. Tu, Y. Yang, J. Zhang, and X. Deng. Flappy Hummingbird: An Open Source Dynamic Simulation of Flapping Wing Robots and Animals. *arXiv:1902.09628 [cs]*, Feb. 2019. arXiv: 1902.09628.

[18] T. Gandhi and M. M. Trivedi. Pedestrian collision avoidance systems: a survey of computer vision based recent studies. In *2006 IEEE Intelligent Transportation Systems Conference*, pages 976–981, Sept. 2006.

[19] G. Hee Lee, F. Faundorfer, and M. Pollefeys. Motion Estimation for Self-Driving Cars with a Generalized Camera. pages 2746–2753, 2013.

[20] J. Heyman. TracTrac: A fast multi-object tracking algorithm for motion estimation. *Computers & Geosciences*, 128:11–18, July 2019.

[21] S. Hinedi. An extended Kalman filter based automatic frequency control loop. 1988.

[22] M. Hua, N. Manerikar, T. Hamel, and C. Samson. Attitude, Linear Velocity and Depth Estimation of a Camera Observing a Planar Target Using Continuous Homography and Inertial Data. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1429–1435, May 2018.

[23] M.-x. Jiang, C. Deng, Z.-g. Pan, L.-f. Wang, and X. Sun. Multiobject Tracking in Videos Based on LSTM and Deep Reinforcement Learning, 2018.

[24] M. Kampelmhler, M. G. Mller, and C. Feichtenhofer. Camera-based vehicle velocity estimation from monocular video. *arXiv:1802.07094 [cs]*, Feb. 2018. arXiv: 1802.07094.

[25] C. Kim, F. Li, and J. M. Rehg. Multi-object Tracking with Neural Gating Using Bilinear LSTM. In V. Ferrari, M. Hebert, C. Sminchisescu, and Y. Weiss, editors, *Computer Vision ECCV 2018*, volume 11212, pages 208–224. Springer International Publishing, Cham, 2018.

[26] Y. Kim, H. Jung, D. Min, and K. Sohn. Deep Monocular Depth Estimation via Integration of Global and Local Predictions. *IEEE Transactions on Image Processing*, 27(8):4131–4144, Aug. 2018.

[27] N. Lee and K. M. Kitani. Predicting wide receiver trajectories in american football. In *2016 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1–9. IEEE, 2016.

[28] A. Lukezic, T. Vojir, L. Cehovin Zajc, J. Matas, and M. Kristan. Discriminative correlation filter with channel and spatial

reliability. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6309–6318, 2017.

[29] P. R. Mahapatra and K. Mehrotra. Mixed coordinate tracking of generalized maneuvering targets using acceleration and jerk models. *IEEE Transactions on Aerospace and Electronic Systems*, 36(3):992–1000, 2000.

[30] A. Milan, L. Leal-Taixe, I. Reid, S. Roth, and K. Schindler. MOT16: A Benchmark for Multi-Object Tracking. *arXiv:1603.00831 [cs]*, Mar. 2016. arXiv: 1603.00831.

[31] S.-i. Nakazawa, T. Ishihara, and H. Inooka. Real-time algorithms for estimating jerk signals from noisy acceleration data. *International Journal of Applied Electromagnetics and Mechanics*, 18(1-3):149–163, 2003.

[32] A. J. Piergiovanni and M. S. Ryoo. Representation Flow for Action Recognition. pages 9945–9953, 2019.

[33] J. A. Prez-Ortiz, F. A. Gers, D. Eck, and J. Schmidhuber. Kalman filters improve LSTM network performance in problems unsolvable by traditional recurrent nets. *Neural Networks*, 16(2):241–250, Mar. 2003.

[34] F. J. Romero-Ramirez, R. Muoz-Salinas, and R. Medina-Carnicer. Speeded up detection of squared fiducial markers. *Image and Vision Computing*, 76:38–47, 2018.

[35] A. Sadeghian, A. Alahi, and S. Savarese. Tracking the Untrackable: Learning to Track Multiple Cues with Long-Term Dependencies. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 300–311, Venice, Oct. 2017. IEEE.

[36] L. Sigal, A. O. Balan, and M. J. Black. HumanEva: Synchronized Video and Motion Capture Dataset and Baseline Algorithm for Evaluation of Articulated Human Motion. *International Journal of Computer Vision*, 87(1-2):4–27, Mar. 2010.

[37] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers. A benchmark for the evaluation of RGB-D SLAM systems. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 573–580, Vilamoura-Algarve, Portugal, Oct. 2012. IEEE.

[38] F. Vasconcelos and N. Vasconcelos. Person-following uavs. In *2016 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1–9. IEEE, 2016.

[39] X. Xia, L. Xiong, W. Liu, and Z. Yu. Automated Vehicle Attitude and Lateral Velocity Estimation Using a 6-D IMU Aided by Vehicle Dynamics. In *2018 IEEE Intelligent Vehicles Symposium (IV)*, pages 1563–1569, June 2018.

[40] I. Zamora, N. G. Lopez, V. M. Vilches, and A. H. Cordero. Extending the OpenAI Gym for robotics: a toolkit for reinforcement learning using ROS and Gazebo. *arXiv:1608.05742 [cs]*, Aug. 2016. arXiv: 1608.05742.

[41] X. Zhang, Y. Wei, J. Feng, Y. Yang, and T. S. Huang. Adversarial Complementary Learning for Weakly Supervised Object Localization. pages 1325–1334, 2018.

[42] A. Z. Zhu, D. Thakur, T. zaslan, B. Pfrommer, V. Kumar, and K. Daniilidis. The Multivehicle Stereo Event Camera Dataset: An Event Camera Dataset for 3d Perception. *IEEE Robotics and Automation Letters*, 3(3):2032–2039, July 2018.

[43] D. Zorbas, T. Razafindralambo, F. Guerriero, et al. Energy efficient mobile target tracking using flying drones. *Procedia Computer Science*, 19:80–87, 2013.