Monocular Video-Based Trailer Coupler Detection using Multiplexer Convolutional Neural Network

Yousef Atoum[1], Joseph Roth[1], Michael Bliss[2], Wende Zhang[2] and Xiaoming Liu[1]
[1]Michigan State University
[2]General Motors
[1]{atoumyou, rothjos1, liuxm}@msu.edu, [2]{michael.j.bliss, wende.zhang}@gm.com

Abstract

This paper presents an automated monocular-camera-based computer vision system for autonomous self-backing-up a vehicle towards a trailer, by continuously estimating the 3D trailer coupler position and feeding it to the vehicle control system, until the alignment of the tow hitch with the trailers coupler. This system is made possible through our proposed distance-driven Multiplexer-CNN method, which selects the most suitable CNN using the estimated coupler-to-vehicle distance. The input of the multiplexer is a group made of a CNN detector, trackers, and 3D localizer. In the CNN detector, we propose a novel algorithm to provide a presence confidence score with each detection. The score reflects the existence of the target object in a region, as well as how accurate is the 2D target detection. We demonstrate the accuracy and efficiency of the system on a large trailer database. Our system achieves an estimation error of 1.4 cm when the ball reaches the coupler, while running at 18.9 FPS on a regular PC.

1. Introduction

Trailers range in size from small utility and boat trailers to large box trailers or recreation vehicle (RV) trailers. RV trailers alone had a revenue of five billion dollars in 2015 [9] and are estimated to exceed 381 thousand units sold in the United States in 2016 [22].

To hitch with a vehicle, trailers have a coupler at the front that is placed over a ball connected to the vehicle at the rear. Small, lightweight trailers may be manually moved into place. However, for heavy trailers, the vehicle must be driven backwards to connect to the stationary trailer. Traditionally, a second person, called a spotter, stands outside the vehicle to instruct the driver. Even with rear-view cameras on modern vehicles, the task is difficult and tedious due to the small size of the coupler on the screen. An automated system can take the place of the spotter and allow a single person to connect to a trailer with ease.

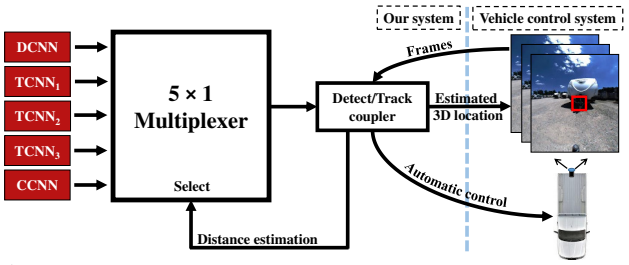

Figure 1: Automatic trailer hitching by detecting and tracking the coupler using a Multiplexer-CNN.

Engineers have developed numerous advanced vehicle systems providing security and comfort for drivers, *e.g.*, emergency breaking, blind spot assist, lane recognition, and active park assist. Current systems for trailers only provide assistance after they are hitched to a vehicle, such as maneuvering in narrow spaces [15], preventing large articulation angles [21], and avoiding oscillation at high speeds [8]. To the best of our knowledge, no other automated system exists for backing-up towards a trailer. This paper presents an automated computer vision system using a single rear-view camera to hitch the vehicle to a trailer, as in Fig. 1.

There are three main challenges or sources of visual variation: position, trailer, and environment. *Position* refers to the pose and scale of the trailer. The *trailer* varies in type, shape, color, and size of the trailer and coupler. The *environment* affects the background of the scene through asphalt, dirt, grass, or snow and the lighting from sun, clouds, or nighttime. Furthermore, there are strong performance requirements needed to successfully hitch the trailer. When hitching, the coupler 2D estimation should have an error less than the radius of the ball, *e.g.,* 2.2 cm for a standard ball. And the efficiency should exceed 10 FPS.

Assuming the driver parks the vehicle with the trailer in the field of view of the camera, this work presents a Multiplexer-CNN based system to automatically detect and track the trailer's coupler and continuously provide 3D coordinates to control a vehicle, while relying solely on a monocular rear-view fish-eye camera. Specifically, the goal of our system is to estimate both the 2D location in *pixels*

and the 3D location in *meters*. An automatic control system may consume the 3D estimate to mechanically control the vehicle's movement. A feedback loop can be established between the control unit and our system, providing additional valuable info for detecting the coupler. However, the mechanical system is outside the scope of this paper.

The Multiplexer-CNN system selects from five CNN architectures to perform detection operations. The current estimate of the coupler position drives the CNN selection. Each CNN is invoked independently based on the estimated distance between the vehicle and the coupler. When no estimate is known, the DCNN detects the coupler by estimating potential locations along with presence confidence scores. We develop a novel loss function to enable DCNN to learn accurate presence confidence measures, along with regression estimates of the coupler position. With a confident estimate of trailer position, the multiplexer selects among three networks, TCNN_1, TCNN_2, and TCNN_3 to perform 2D tracking by detection until the coupler is centered over the ball. During this time, the 3D location is inferred using a calibrated distance map from a fixed coupler height. As the coupler approaches the ball, it is crucial to estimate the height to avoid collision. The fifth CNN, CCNN, estimates the coupler's contour, which regresses the height, and adjusts the distance map to provide accurate 3D estimation.

Data is key to learning an accurate Multiplexer-CNN. We introduce the first large-scale dataset for trailers, with 899 videos containing $\sim 712,000$ frames. We demonstrate the accuracy and efficiency of the system using a regular PC. Our system fulfills the performance requirements for successful hitching by achieving an estimation error of 1.4 cm when the ball reaches the coupler, while running at 18.9 FPS. Qualitatively, we show the ability of our system to detect and track unseen trailers in the field, generalizing to handle a large variety of challenges.

In summary, our main contributions are: 1) Develop a novel loss function along with a CNN-based object detector, which estimates coupler coordinates associated with presence scores. 2) Design a distance-driven Multiplexer-CNN that achieves both generalization across larger variations and real-time efficiency. 3) Develop a method to estimate the 3D coupler coordinate with a monocular camera. 4) Present a large dataset for trailer detection and tracking.

2. Related Work

We review prior work in three areas: object detection, object tracking, and 3D localization.

Object detection Recent approaches using CNNs to regress the location of object bounding boxes demonstrate remarkable accuracy and efficiency. Sermanet et al. [26] propose a regression network for detection where classification confidence helps aggregate proposed bounding boxes. However, it exhaustively searches the image and is not suitable for real-time applications. Girshick et al. [11] propose R-CNN, using object proposals generated by selective search. This model has been extended to Fast R-CNN [10] and Faster R-CNN [25]. In Faster R-CNN, the region proposal network (RPN) generates a set of rectangular object proposals, each with an objectness score measuring membership to an object class, after which the proposal is assigned a class specific confidence score. Note that both the objectness score of RPN and classification confidence are trained separately from localization and are thresholded to produce a binary detection decision. In contrast, we incorporate the presence confidence score into the learning process and use it as a scalar without thresholding. Our score reflects two major cues: (1) the existence of the target object in a region, and (2) how accurate is the 2D target detection.

Object tracking As shown in a recent benchmark study [6], Correlation Filters (CF) [14, 19, 13] and CNNs [16, 29, 32] achieve the top performances. CFs provide better efficiency, while CNNs have superior accuracy. Both approaches work well on the scale variation observed in benchmarks. However, for trailer videos, the extreme scale variation induces substantial appearance variation due to changes in perspective that current single object trackers cannot handle. We propose multiple CNN trackers, *i.e.*, TCNN_i, for accurate tracking at all scales and compare against state-of-the-art single trackers.

3D Object localization Recently, many works aim to localize objects in 3D for autonomous vehicles [31, 12, 3, 4]. RADAR and LIDAR [12, 4] are popular sensors. With two cameras, stereo vision [3] provides depth. Many obstacle detection systems use stereo vision, *e.g.*, the occupancy map [2] and digital elevation map [23]. However, vehicles come equipped with a *monocular backup camera*.

Structure from Motion (SfM) uses a monocular camera to localize a moving camera and objects [17]. However, SfM was shown to have difficulties in real driving scenarios due to the small object size, fast speeds and lack of texture. On the other hand, [27] performs 3D object localization by combining SfM cues such as the ground plane, with object cues such as 2D bounding boxes. We assume the height of the camera is fixed relative to the ground and create a distance map offline. Then the height of the coupler, estimated via the 3D coupler localization algorithm, is used to dynamically elevate the distance map for efficient localization.

3. Our Proposed Approach

Our Multiplexer-CNN system has five CNN inputs: DCNN, TCNN_1, TCNN_2, TCNN_3, CCNN. As shown in Fig. 2, our system consists of three stages: (1) 2D coupler detection, (2) 2D coupler tracking, and (3) 3D coupler localization for vehicle automation. Stage 1 initializes the 2D coordinate of the coupler for Stage 2. Stage 2 and 3 collaborate in estimating both 2D and 3D coupler positions.

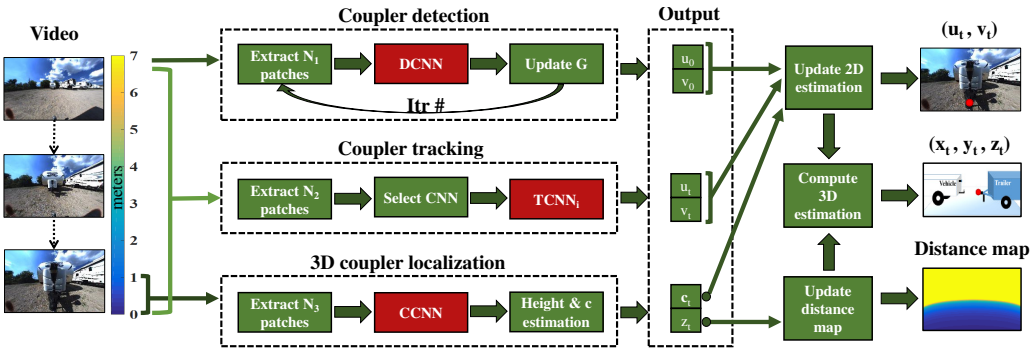

Figure 2: An automated computer vision system for coupler detection, tracking, and 3D localization.

3.1. Preprocessing for Geometric Interpolation

Our rear-view camera has a fish-eye lens with a wide field-of-view (FOV). Using a checkerboard, standard camera calibration is performed to estimate camera intrinsics, extrinsics, and lens distortion parameters, based on which we unwarp the input frame to correct the fish-eye effect.

In our Multiplexer system, the coupler-to-vehicle distance (CVD) is crucial in serving as the selector to choose one CNN among five. Thus, it is important to estimate CVD accurately and efficiently. Instead of employing SfM for distance estimation [27], we propose to rely on a distance map covering the entire frame. This distance map can be obtained as follows. Given an unwarped frame with a checkerboard placed on a flat ground, we estimate the camera rotation and translation matrices, which can convert a single pixel (u, v) to a 3D world coordinate and measure the distance between the camera and target pixel (green point in Fig. 3). This step is repeated for all pixels in the unwarped frame. Given the known (fixed) camera height h_0, solving a simple triangle problem can convert the camera-to-pixel distance to the origin-to-pixel distance, where the origin of the 3D world coordinate is the projection of the camera onto the ground plane ($(0, 0, 0)$ in Fig. 3). The origin-to-pixel distances of all pixels constitute the distance map on the ground plane $\mathbf{D}_0$. Finally, given the coupler is at a fixed but unknown height h_c, we elevate $\mathbf{D}_0$ to obtain the coupler distance map $\mathbf{D}_h$ by simply applying $\mathbf{D}_h = (1 - \frac{h_c}{h_0})\mathbf{D}_0$. In our system, we assume $h_c = 50$ cm when the CVD is over 1 meter, and otherwise estimate using the method in Sec. 3.4. For different vehicles with different h_c, the camera should be recalibrated to generate a new ground distance map $\mathbf{D}_0$. We can do this for all possible heights offline, and assign one to a vehicle during manufacture.

Our distance map method has a few advantages. First, we only use a single camera without additional sensors/cameras to retrieve 3D depth information. Second, $\mathbf{D}_h$ can be updated efficiently to accommodate changes in the coupler height. Finally, CVD is obtained by a simple look-up of $\mathbf{D}_h(u, v)$. However, our flat ground assumption might affect CVD estimation if the ground is not flat. Fortunately,

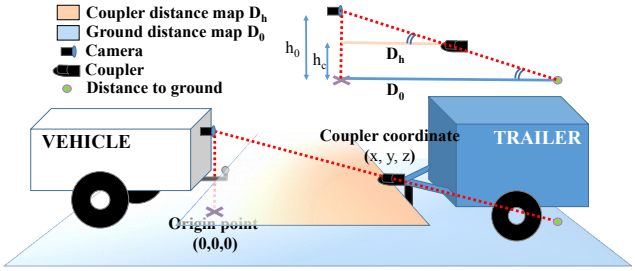

Figure 3: 3D distance estimation of the coupler.

this would impact most when the trailer is far away, and have much less influence while approaching it, since close regions become locally flat. Note that the critical distance is the last CVD meter when the coupler is near the vehicle.

3.2. Coupler Detection

When the system starts, we assume the trailer is positioned between three and seven meters from the rear of the vehicle. This is a reasonable assumption for our application. If the trailer starts too close, the vehicle may not have enough room to align with the hitch. If too far away, the coupler cannot be detected. We train a dedicated network, DCNN, to detect the coupler in the frame along with providing a presence confidence score of the estimation.

DCNN network The architecture of DCNN is in Tab. 1. Each convolution layer is followed by an ReLU and max-pool layer. The key to DCNN is the novel *presence loss* that allows the detector to balance the regression accuracy with the detection presence confidence accuracy. These scores reflect the existence of the target object within the spatial enclosure of the training patches, as well as how accurate the 2D regression detection. The *presence loss* is,

$$L = \sum \bar{s}||\Delta\bar{\mathbf{u}} - \Delta\mathbf{u}||^2 + \sum \lambda_1||s - 2\bar{s}(1 - \text{sigm}(\lambda_2||\Delta\bar{\mathbf{u}} - \Delta\mathbf{u}||^2))||^2, \quad (1)$$

where $\Delta\mathbf{u}$ and s are the estimated 2D offset and presence confidence score, and $\Delta\bar{\mathbf{u}}$ and $\bar{s}$ are the groundtruth. The loss function has two parts. The first part represents the Euclidean loss, which is the regression error when having a

Table 1: CNN architectures.

DCNN	$\text{TCNN}_{1,2,3}$	CCNN
input(200×200)	input(224×224)	input(200×200)
conv ($7 \times 7 \times 20$)	conv ($3 \times 3 \times 20$)	conv ($7 \times 7 \times 20$)
maxpool (2)	conv ($3 \times 3 \times 20$)	maxpool (3)
conv ($7 \times 7 \times 30$)	maxpool (3)	conv ($5 \times 5 \times 40$)
maxpool (2)	conv ($3 \times 3 \times 40$)	maxpool (3)
conv ($5 \times 5 \times 40$)	conv ($3 \times 3 \times 40$)	FC-100
maxpool (2)	maxpool (3)	FC-76
FC-100	conv ($3 \times 3 \times 60$)	Euclidean loss
FC-3	maxpool (2)	—
Presence loss	conv ($3 \times 3 \times 80$)	—
—	maxpool (2)	—
—	conv ($3 \times 3 \times 100$)	—
—	FC-100	—
—	FC-2	—
—	Euclidean loss	—

positive coupler patch enabled by $\bar{s}$. For negative patches where $\bar{s} = 0$, loss is ignored so learning focuses only on the presence confidence score. The second part is the presence confidence score loss. For negative patches, it penalizes nonzero scores since the system should have no confidence. For positive patches, the confidence should be negatively correlated to the regression error, *i.e.*, more accurate predictions of $\Delta\mathbf{u}$ should have higher presence confidence. We pass the regression error through a sigmoid function and subtract from 1, restricting confidence between 0-1.

We obtain training image patches to learn DCNN. The positive patches are obtained from the videos when the trailer is in the range of d_1 to d_2, and are assigned confidence scores $\bar{s} = 1$, along with the coupler offset $\Delta\bar{\mathbf{u}}$. Random perturbation is applied to the patches such that the center of the coupler is less than $|\Delta\bar{\mathbf{u}}| < \frac{w}{4}$ away from the patch center, where w is the patch size. The negative patches are randomly selected in the surrounding area of the coupler such that the coupler center is far away from the patch center $|\Delta\bar{\mathbf{u}}| > \frac{w}{4}$, and are assigned scores of zero.

Coupler detection algorithm An illustration of the coupler detection algorithm is in Fig. 2. Given the initial trailer location within $[d_1, d_2]$ meters, we place a grid G containing N_1 evenly distributed points covering the region as shown in the first column of Fig. 4. These points represent the center locations of N_1 test patches, which are processed by DCNN. These patches have different sizes based on their estimated CVD d_t, such that the patches in the top row of G have the smallest sizes, whereas the bottom row have the largest sizes. This is motivated by the desire to feed CNN training data with smaller scale variations. Otherwise if we would select a fixed patch size, *e.g.*, 200×200, the patches contain a lot of background info when the trailer is far away, and nearly no background when near by. We follow a simple formula to decide the patch size via CVD,

$$w = \frac{-45}{2}\mathbf{D}_h(u_{t-1}, v_{t-1}) + 300. \quad (2)$$

Here the maximum patch size is 300×300 when the trailer is at zero meters away, and 75×75 when 10 meters away.

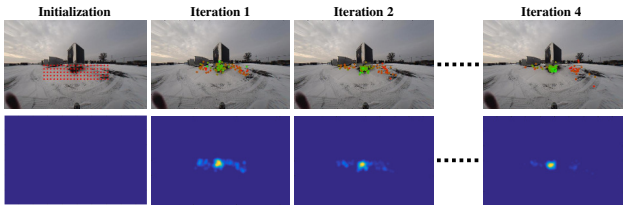

Figure 4: Iterative coupler detection using DCNN. The top row is the input image with the results of ($\Delta\mathbf{u}$, s). The green color means $s = 1$, and the red is $s = 0$. The bottom row shows the results of the weighted sum of Gaussians. Best viewed in color.

Once the scale is determined, the patches are resized to 200×200. Since many patches overlap with the target coupler, it is likely that a number of patches will have high presence scores. We update each grid point with its estimated location and repeat the process iteratively on the same initial frame, until the points cluster around potential couplers.

In this particle filter like approach, we adopt a weighted sum of Gaussians approach for a final detection estimation. Every grid point is replaced with a 2D Gaussian (100×100 kernel size with $\sigma = 30$) weighted by the presence score. The final detection is the maximum of the summation of weighted Gaussians from all grid points. The estimated presence scores of correct points increases with more iterations performed. In general, we observe that four iterations are sufficient for satisfactory accuracy as seen in Fig. 4.

3.3. Coupler Tracking

In coupler tracking, the coupler appearance changes throughout the backing-up process, which can be described in three stages: (a) Initially, the coupler is far and often hard to discern. (b) As the trailer gets closer, the coupler appears increasingly larger. (c) Within the last meter of backing-up, the coupler appearance changes dramatically due to the increasing downward viewing angle of the camera. Therefore, we propose to use three networks, TCNN_1, TCNN_2 and TCNN_3, to perform tracking by detection.

Tracking CNN networks The network architecture of TCNN is in Tab. 1. The first 10 layers are similar to the VGG network [24], with minor changes in the number of filters and maxpool layers. The full network is optimized using training images of trailer couplers. Unlike most object tracking works estimating a bounding box, we are only interested in finding the center of the coupler. Therefore, we define the Euclidean loss for the 2D coupler center position.

A large number of training images are used to learn all three TCNN networks. Similar to DCNN, we apply random perturbation to the training patches such that the coupler center is less than $\frac{w}{4}$ away from the patch center. We also follow Eqn. 2 to crop the local region with the CVD-dependent size to a training patch.

Coupler tracking algorithm We adopt a tracking by detection method where TCNN serves as the detector. We use the tracking result of the previous frame to initialize the

Algorithm 1: 3D coupler localization algorithm

Data: Input frame $\mathbf{I}_t$, (u_t,v_t), z_{t-1}, $\mathbf{D}_{h,t-1}$
Result: (x_t,y_t,z_t), $\mathbf{D}_{h,t}$

Obtain current distance $d_t = \mathbf{D}_h(u_{t-1}, v_{t-1})$;
if $d_t > \tau_3$ **then**
  $z_t = 0.5$ m, compute (x_t, y_t) using Fig. 5b;
else
  Multiplexer selects CCNN;
  for $i := 1$ *to* N_3 **do**
    Generate random perturbation r_u^i and r_v^i;
    Generate patch $\mathbf{P}_i = \mathbf{I}_t(u_t + r_u^i, v_t + r_v^i; d_i)$;
    $\mathbf{c}_i \longleftarrow$ CCNN$(\mathbf{P}_i)$;
    $\mathbf{b} = \mathbf{P}^T(\mathbf{c}_i - \bar{\mathbf{c}}_\mathbf{0})$;
    if $\neg$ *ValidShapeTest*($\mathbf{b}$) **then**
      remove $\mathbf{P}_i$ and $\mathbf{c}_i$;
  Mean of all contours $\mathbf{c} = \frac{1}{\hat{N}_3}\sum_i^{\hat{N}_3} \mathbf{c}_i$;
  Feature extraction $\mathbf{f} = F(\mathbf{c})$ using Fig. 5a;
  Compute $z_t = R_j(\mathbf{f})$; // j is Regressor index;
  Update $\mathbf{D}_{h,t} = \mathbf{D}_0(1 - z_t/h_0)$;
  Update $(u_t, v_t) = \frac{1}{2}(u_t, v_t) + \frac{1}{2}(\mathbf{c}(75), \mathbf{c}(76))$;
  Update (x_t, y_t, z_t) using Fig. 5b;

current frame. For stable tracking, we apply N_2 randomly perturbed patches surrounding the initialization. The tracking result is obtained by averaging the estimations of all N_2 patches. Two thresholds, τ_1 and τ_2, define three ranges where each of the three TCNNs operate.

3.4. Height Estimation and 3D Localization

The motivation of estimating the coupler height is twofold. 1) To control the vehicle mechanically, where the vehicle control system requires a precise 3D location of the coupler, (x, y, z) which is the range, offset and height in meters. This demands the distance map at the true height of the coupler. Hence, we need to estimate the coupler height rather than using the assumed height. 2) The vehicle has a hitch ball set to a fixed height. We need to ensure that the coupler is high enough to avoid colliding with the hitch ball.

It is challenging to estimate the coupler height from a monocular camera. For fixed size objects like license plates, the depth can be estimated from the plate pixel size [5]. However, couplers vary significantly in their shape. To address this problem, we discover that the geometric shape of the coupler contour is indicative of the height, *e.g.*, at a fixed CVD, increasing the height of a coupler will spread out the contour points.Therefore, we propose to estimate the coupler contour using CCNN. This allows us to extract contour geometric features and feed them to regressors to estimate coupler heights, as detailed in Alg. 1.

Coupler contour network The network architecture of CCNN is in Tab. 1. Given the small number of contour labels, we learn a shallow network of 2 convolution layers and 2 FC layers, with an ReLU and maxpool layer after each convolution layer. CCNN defines the Euclidean loss on the coupler contour represented by a 76-dim $\mathbf{c}$, *i.e.*, 38 points, where 37 points are on the contour, and the last one $(\mathbf{c}(75), \mathbf{c}(76))$ is the coupler center.

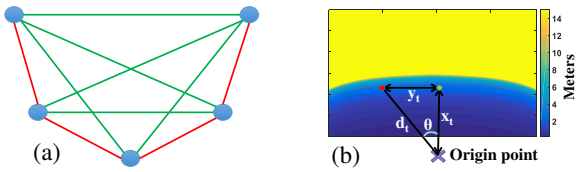

Figure 5: (a) Geometric feature of a contour include: distances along straight lines, and slopes of red lines. (b) Estimation of x_t and y_t, on the $\mathbf{D}_h$ elevated by estimated height z_t. Red dot is the estimated coupler, green has zero offset from the origin point.

Contour estimation Similar to tracking, to improve estimation stability, our contour is estimated using candidate contours of N_3 patches extracted with random perturbations, each obtained by CCNN. Due to the high-dimension output, there are normally a few outliers among the candidate contours. Therefore, we propose to use a shape model [20] to remove the outliers, *i.e.*, contours with unusual coupler shapes. Based on labeled contour training images $\bar{\mathbf{c}}$, we compute a mean shape $\bar{\mathbf{c}}_\mathbf{0}$ and five basis shapes $\mathbf{P}$, such that any candidate contour $\mathbf{c}$ can be represented as a coefficient $\mathbf{b} = \mathbf{P}^T(\mathbf{c} - \bar{\mathbf{c}}_\mathbf{0})$. If any candidate contour's $\mathbf{b}$ does not meet the normal coefficient distribution learned from training data, the contour will be ignored when making a final mean estimation.

Height estimation Given a stable contour estimation, we extract geometric features to capture the 2D shape of the coupler as shown in Fig. 5a. Specifically, we uniformly sample five points along the contour, including the two end points. We compute the Euclidean distance between any two points, resulting in 10 features, *i.e.*, red and green lines. We further compute the slopes of the red lines, resulting in four features. Thus, a 14-dim feature vector is extracted from the contour estimation of CCNN. Given five sets of training images, each having couplers at a specific CVD, we utilize their feature vectors to learn five height estimators $\{R_i\}_{i=1}^5$ via the bagging M5P regressor [30, 1]. Our analysis shows bagging M5P to be superior to other well-known regression paradigms.

3D coupler localization A detailed algorithm for 3D coupler localization is in Alg. 1. Given the current 2D coupler location (u_t, v_t) estimated via TCNN, we find the CVD d_t utilizing the distance map $\mathbf{D}_h$ at the assumed height of 50 cm. However, when d_t is less than $\tau_3 = 1$ meter, CCNN is activated to estimate the contour, followed by the height estimation for each video frame. Then a refined CVD d_t can be retrieved for two updates: First, the distance map $\mathbf{D}_h$ is elevated to the estimated coupler height h_c. Second, the 2D coupler location (u_t, v_t) is refined by averaging the TCNN

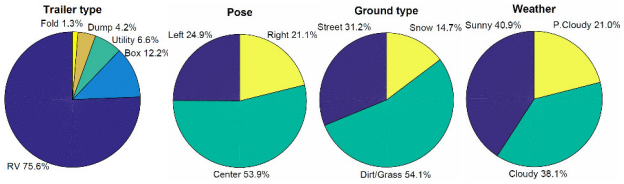

Figure 6: Statistics of the trailer coupler database.

result with the coupler center estimation from CCNN.

Independent of whether d_t is less than τ_3, we need to convert the CVD d_t to the 3D coupler localization (x_t, y_t, z_t), for the purpose of vehicle control. To find x_t and y_t, we solve a simple triangle problem on the distance map at the coupler height, where x_t and y_t are the two sides forming the 90° angle as seen in Fig. 5b, and the third side is the CVD d_t. As for z_t, it is either the assumed height of 50 cm if $d_t > \tau_3$ or otherwise the estimated height h_c.

4. Trailer Coupler Database

With a rear-view camera of a vehicle, the videos capture the process of a vehicle backing-up towards a trailer from a variable distance to the point where the hitch ball is aligned with the trailer coupler. The database contains 899 videos consisting of $\sim 712K$ frames, with an average length of 19.3 seconds. The videos are collected at 40 FPS with a resolution of $1,920 \times 1,200$ using M-JPEG compression. A Point Gray camera (Model: BFLY-PGE-23S6C-C) with a fish-eye lens is used with a wide FOV of 190°. The database contains three types of challenges as explained in Sec. 1. Fig. 6 shows some statistics of the database. Nearly $\frac{3}{4}$ of the database is obtained from RV trailers, as they are the most popular and available. Even if the trailer type introduces differences in shape and size, they all have similarities in the coupler itself. Typically, people backup to the trailer in a straight line with a center pose, however, we collect videos with various poses to make the system robust to any pose situation. The database is collected in a period of one year, which covered different weather conditions and ground types. Some examples can be seen in Fig. 15.

There are three types of labels in the dataset. 1) Coupler label set: the 2D coordinate of the coupler center is labeled at every 10th frame of all 899 videos. 2) Contour label set: the 2D coordinates of 37 coupler contour points are labeled for a set of 810 frames, collected from 162 videos, when the CVD is 1.0, 0.8, 0.6, 0.4, and 0.2 meters away. 3) Height label set: the coupler height is physically measured at the site when 72 videos are captured. These 72 videos are a subset from the 162 videos of the contour label set, which means we also have contour labels along with them.

5. Experiments

In this section, we will discuss the experimental setup, report the quantitative results of all three main stages separately, and then jointly as a whole system. Finally, we will show qualitative results of the entire system.

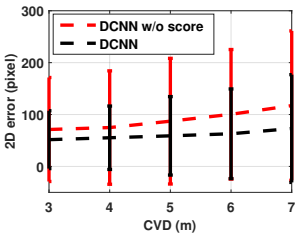

Figure 7: Detection errors vs with distance in meters.

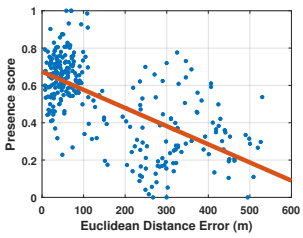

Figure 8: Presence scores vs Euclidean estimation errors.

5.1. Multiplexer-CNN Setup

Network Implementation Details We train and test DCNN and TCNN using the first 827 videos using only the coupler label set. The videos are divided into 413 training videos, and 414 testing videos. Given most trailers have $2\sim3$ videos captured at different posses, we make sure that each unique trailer does not exist in both training and testing sets. To train and test CCNN, we use all videos which contained contour labeled frames. The videos are divided into 90 training videos with 450 labeled coupler contour frames, and 72 testing videos with 360 labeled coupler contour frames. The networks are trained with a learning rate of 0.001 and a mini-batch size of 100, 20, and 20 for DCNN, TCNN and CCNN, respectively.

Parameter setting Our system uses the following parameters: $\tau_1 = \tau_2 = 3$, $\tau_3 = 1$, $N_1 = 100$, $N_2 = N_3 = 10$, $\lambda_1 = \lambda_2 = 1$. For $ValidShapeTest(\mathbf{b})$ in Alg. 1, it returns true if all elements of $\mathbf{b}$ are within three standard deviations of coefficient distributions. The five height estimators $\{R_i\}_{i=1}^5$ are trained from contours at distance ranges of $(0.9, 1.1]$, $(0.7, 0.9]$, $(0.5, 0.7]$, $(0.3, 0.5]$, $(0.1, 0.3]$ meters, respectively.

5.2. Results

Coupler detection For coupler detection, the baseline is similar to DCNN with the exact same network structure except for the loss function, which is a normal Euclidean loss, *i.e.*, it only estimates $\Delta\mathbf{u}$ without the presence confidence score s. Hence, the coupler detection algorithm remains the same, except replacing the sum of weighted Gaussians with a sum of Gaussians. Fig. 7 reports the results of both methods at various initializations of CVD in the range of $3\sim7$ meters. This experiment shows the advantage of learning presence confidence score over the typical regression detectors. A clear margin of $20\sim45$ in pixel errors is due to incorporating presence confidence score learning into DCNN.

To visualize the effectiveness of our presence confidence scores in DCNN, we demonstrate the correlation between the estimated scores and the 2D offset estimation errors. Given six randomly selected testing videos, we collect a total of 600 pairs of $(\mathbf{u}, s)$ after running DCNN at seven CVD meters for four iterations. As seen in Fig. 8, the scores have an inverse correlation with the Euclidean estimation errors, *i.e.*, the detections with lower errors will have higher scores.

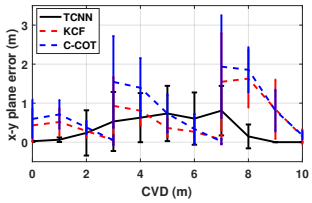

Figure 9: Tracking comparison with errors in meters.

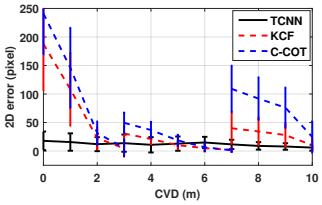

Figure 10: Tracking comparison with errors in pixels.

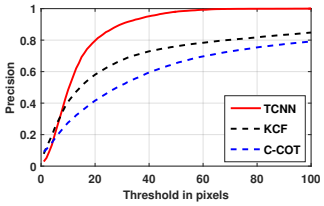

Figure 11: Tracking comparison with precision plot.

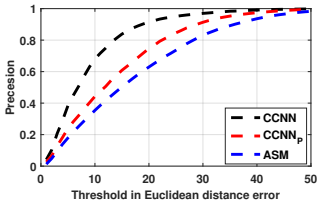

Figure 12: Contour estimation comparison.

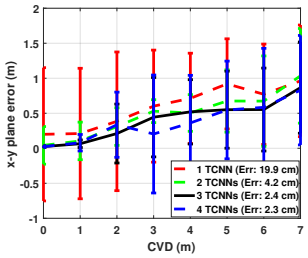

Figure 13: Number of TCNN networks used for tracking.

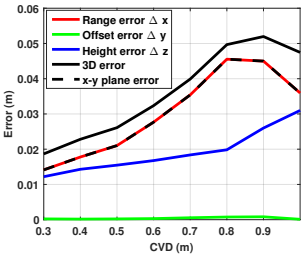

Figure 14: Overall system accuracy on the height label set.

The correlation coefficent is found to be -0.65 capturing this strong correlation between these two. Note that some large offset estimation errors might still have a high score. This false alarm is caused by the patch drifting away from the coupler, where it detects some background object thinking it's the target coupler. An example is found in iteration 1 of Fig. 4, where some points illustrate this noisy behavior.

Coupler tracking We compare our TCNN with two baseline object tracking methods, KCF [13] and C-COT [7]. Both methods have excelled on many tracking benchmarks, *e.g.*, C-COT wins the 2016 VOT challenge. To compare with the baseline, we provide three initializations to both baselines, while one initialization to our method. For the baseline tracking initialization, we define a 100×100 bounding box centered around the coupler, at three CVD locations d_0, $d_1 = 7$ and $d_2 = 3$ meters. Here d_0 is the CVD of the first video frame, with an average of 10 meters in our database. We report resultant $x - y$ plane error in meters at specific CVD in Fig. 9, 2D pixel errors at specific CVD in Fig. 10, and precision plots in Fig. 11. We observe that both KCF and C-COT perform well after initialization. However, both baseline methods suffer from drifting problems due to the extreme scale variations, which induces substantial appearance variation caused by changes in perspective. To the best of our knowledge, this challenge is rarely addressed in any of the tracking benchmark studies.

Using three TCNN networks was specifically justified due to large appearance variation in the trailer coupler as the CVD converges to zero meters. We have experimentally studied the effect of using different numbers of TCNN networks, as seen in Fig. 13, with the final error at zero CVD meters in parenthesis. With increasing number of TCNN networks, a higher tracking accuracy can be achieved yet system complexity increases. Thus, we choose three TCNN networks to balance the accuracy and complexity.

Table 2: System efficiency.

System	DCNN	TCNN	CCNN	Alg 1	Alg 2	Alg 3
Time(s)	0.088	0.023	0.010	0.471	0.053	0.035

Height estimation Contour estimation is crucial to our height estimation. We first compare our CCNN-based contour estimation with two baselines, ASM contour fitting [20], and CNN-based polynomial coefficient fitting inspired by [18], which used curve functions to describe the facial contour. For the classic ASM method, we learn a 2D ASM model to iteratively fit a shape to the coupler contour. For the second baseline, we learn a CNN named CCNN_P similar to CCNN with the same structure, except that instead of producing a regression output for 38 contour points, CCNN_P estimates an 8-dim vector (α_i, β_i) for $i = 1 \cdots 4$. Here α_i and β_i are the third degree polynomial coefficients of the x-coordinate and y-coordinate of the contour points, respectively. We report the Euclidean distance error by measuring the shortest path of the estimated point to the groundtruth contour in Fig. 12. Observing a fixed threshold at 20 pixels, our CCNN has a precision rate of 92% compared to 75% and 63% for CCNN_P and ASM.

Given the coupler contour and its geometric features, we can estimate the height. We analyze the performance of the five height estimators through a 5-fold cross validation on the 72 videos of the height labeled set. The absolute mean errors are 0.85, 0.75, 0.60, 0.54, and 0.45 cm for $\{R_i\}_{i=1}^{5}$. We observe higher performance for both contour and height estimation the closer we get to the coupler.

Given the estimated height, we adjust the distance map, based on which we retrieve the CVD using the coupler center estimated by CCNN, *i.e.*, $(\mathbf{c}(75), \mathbf{c}(76))$. Then we perform the 3D coupler localization using the same triangulization in Fig. 5b. The blue curve in Fig. 14 shows the height estimation accuracy using the height label set of 72 videos.

Overall system test We report the results of the entire system, using components from all three stages, in Fig. 14. We use the height label set, because the labeled height (*i.e.*, its elevated distance map) and the labeled coupler center can provide the ground truth 3D coupler locations in each video frame. Note that the 2D coupler center is estimated by fusing TCNN_3 and CCNN, as in Line 17 of Alg. 1. The minimum error can be found in the offset estimation, followed by the height and range estimations. The small error of off-

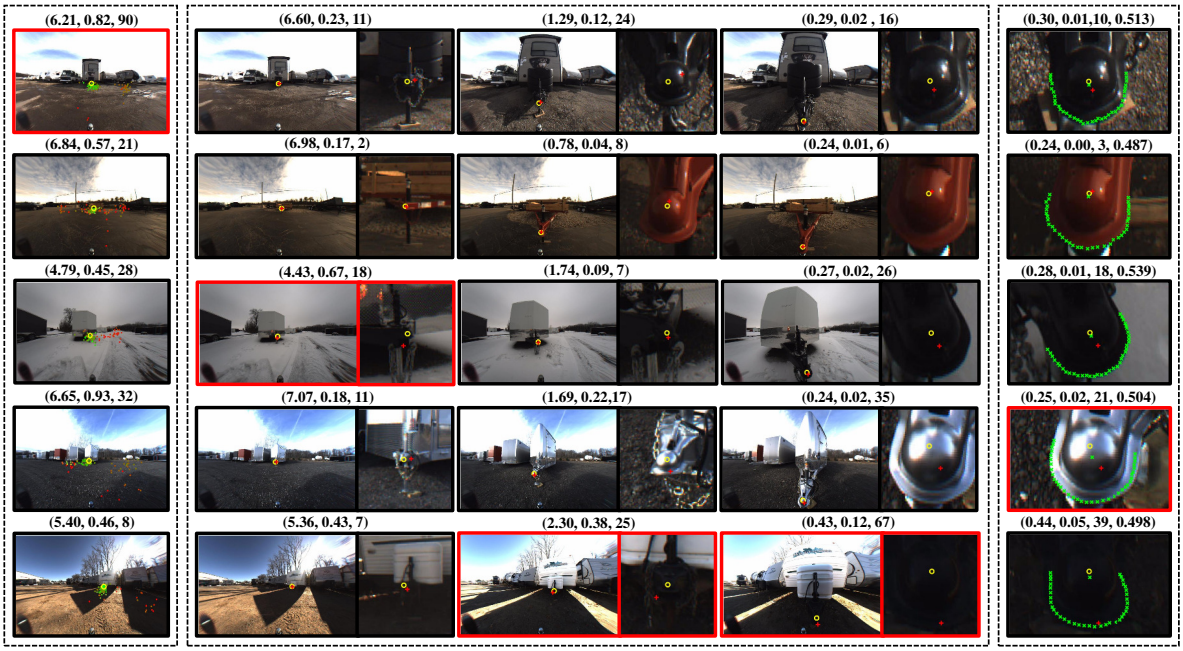

Figure 15: Qualitative results of five videos. The first column is the DCNN results, the three middle columns are TCNN results, last column is the CCNN result. Red + is the estimated result, yellow ○ is groundtruth, green × is the estimated contour. Above each frame is (CVD, meter error, pixel error), the last column also has the estimated height in meters. The red rectangles indicate the failure cases.

set is due to the fact that most backups are along the frontal angle, and therefore, the error on the $x-y$ plane is almost the same as the range error. The final estimation error on the $x-y$ plane is merely 1.4 cm. While not directly comparable due to different datasets, it is substantially smaller than TCNN_3 error of 2.4 cm in Fig. 9. This $x-y$ plane error is the most important accuracy metric for our system. At 1.4 cm, the vehicle control system would drive the vehicle to park and stop at a point where the hitch ball is only 1.4 cm away from the coupler center. The fact that most couplers have a radius of 2.2 cm means that users can easily lower the coupler and hook up with the hitch.

System efficiency Our system is implemented in MATLAB using MatConvNet [28] on an Intel Core i7 $-$ 4770 CPU with 3.40GHz and a single NVIDIA TITAN X GPU. The system can run in real-time obtaining frames from the rear-view fish-eye camera, or offline using the trailer coupler database for analysis. Table. 2 provides detailed efficiency analysis per CNN network and per system stage, where Alg 1, Alg 2, and Alg 3 are the detection, tracking and localization of the coupler when tested with N_1 (for 1 iteration), N_2 and N_3 patches. Note that Alg 1 requires nearly 1.88 seconds to perform 4 iterations during initialization of the system. While tracking the coupler, we can achieve 18.9 FPS. However, when the CVD passes τ_3 the system drops in speed to 11.4 FPS due to running both TCNN and CCNN.

Qualitative results Figure 15 shows qualitative results of detecting and tracking the coupler for full video sequences. We illustrate five different video examples, representing typical challenging cases in the database. We show a few failure cases of the system obtained from different stages of the videos. Note that the fourth and fifth columns use the same frame to illustrate the results of TCNN and CCNN, respectively. One key observation in our system, is that our Multiplexer-CNN approach has the ability to overcome failure cases at various stages of the system. *E.g.*, if our TCNN fails within the last few frames, such as the last row of Fig. 15, our CCNN has a good chance of correcting the problem. The same observation is made when Multiplexer-CNN switches between any two possible networks.

6. Conclusions

We present an automated computer vision system for backing-up a vehicle towards a trailer, using a distance-driven Multiplexer-CNN. While relying solely on a monocular rear-view fish-eye camera, we are able to provide accurate 3D coordinates of the coupler, which are needed for vehicle control. One of the key contributions in this system, is the ability to detect the trailer using DCNN through a new loss function producing presence confidence measures associated with regression estimates. Our quantitative and qualitative results on the collected large-scale trailer database, demonstrate our system's ability to be integrated in any vehicle with a rear-view camera. This work also represents how successful vision systems are built to meet real world needs. From the technical perspective, other applications such as autonomous driving problems, may benefit from our components, e.g., presence confidence loss function, distance map, Multiplexer-CNN.

References

[1] Y. Atoum, M. J. Afridi, X. Liu, J. M. McGrath, and L. E. Hanson. On developing and enhancing plant-level disease rating systems in real fields. *Pattern Recognition*, 53:287–299, 2016. 5

[2] H. Badino, U. Franke, and R. Mester. Free space computation using stochastic occupancy grids and dynamic programming. In *Proc. Int. Conf. Computer Vision Workshops (ICCVW)*, volume 20, 2007. 2

[3] N. Bernini, M. Bertozzi, L. Castangia, M. Patander, and M. Sabbatelli. Real-time obstacle detection using stereo vision for autonomous ground vehicles: A survey. In *Proc. IEEE Int. Conf. Intelligent Transportation Systems (ITSC)*, pages 873–878. IEEE, 2014. 2

[4] M. Birdsall. Google and ITE: the road ahead for self-driving cars. *Institute of Transportation Engineers. ITE Journal*, 84(5):36–39, 2014. 2

[5] S.-H. Chen and R.-S. Chen. Vision-based distance estimation for multiple vehicles using single optical camera. In *Int. Conf. Innovations in Bio-inspired Computing and Applications (IBICA)*, pages 9–12. IEEE, 2011. 5

[6] Z. Chen, Z. Hong, and D. Tao. An experimental survey on correlation filter-based tracking. *arXiv preprint arXiv:1509.05520*, 2015. 2

[7] M. Danelljan, A. Robinson, F. S. Khan, and M. Felsberg. Beyond correlation filters: learning continuous convolution operators for visual tracking. In *Proc. European Conf. Computer Vision (ECCV)*, pages 472–488. Springer, 2016. 7

[8] C. de Saxe and D. Cebon. A visual template-matching method for articulation angle measurement. In *Int. Conf. Intell. Transportation Systems*, pages 626–631. IEEE, 2015. 1

[9] C. for Economic Vitality. n.d. Revenue of the U.S. RV park industry from 2009 to 2015. *Statista*, 2016. 1

[10] R. Girshick. Fast R-CNN. In *Proc. Int. Conf. Computer Vision (ICCV)*, pages 1440–1448, 2015. 2

[11] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, pages 580–587, 2014. 2

[12] J. Han, D. Kim, M. Lee, and M. Sunwoo. Enhanced road boundary and obstacle detection using a downward-looking lidar sensor. *IEEE Trans.Vehicular Technology*, 61(3):971–985, 2012. 2

[13] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista. High-speed tracking with kernelized correlation filters. *IEEE Trans. Pattern Anal. Mach. Intel.*, 37(3):583–596, 2015. 2, 7

[14] Z. Hong, Z. Chen, C. Wang, X. Mei, D. Prokhorov, and D. Tao. Multi-store tracker (MUSTer): A cognitive psychology inspired approach to object tracking. In *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, pages 749–758, 2015. 2

[15] B. Li and Z. Shao. Precise trajectory optimization for articulated wheeled vehicles in cluttered environments. *Advances in Engineering Software*, 92:40–47, 2016. 1

[16] H. Li, Y. Li, and F. Porikli. DeepTrack: Learning discriminative feature representations by convolutional neural networks for visual tracking. In *Proc. British Machine Vision Conf. (BMVC)*, 2014. 2

[17] T. Li, V. Kallem, D. Singaraju, and R. Vidal. Projective factorization of multiple rigid-body motions. In *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, pages 1–6. IEEE, 2007. 2

[18] X. Liu, Y. Tong, F. W. Wheeler, and P. H. Tu. Facial contour labeling via congealing. In *Proc. European Conf. Computer Vision (ECCV)*, pages 354–368. Springer, 2010. 7

[19] C. Ma, X. Yang, C. Zhang, and M.-H. Yang. Long-term correlation tracking. In *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, pages 5388–5396, 2015. 2

[20] I. Matthews and S. Baker. Active appearance models revisited. *Int. J. Computer Vision*, 60(2):135–164, 2004. 5, 7

[21] J. Morales, A. Mandow, J. L. Martinez, J. L. Martínez, and A. J. García-Cerezo. Driver assistance system for backward maneuvers in passive multi-trailer vehicles. In *Proc. Int. Conf. Intelligent Robots and Systems (ICIRS)*, pages 4853–4858. IEEE, 2012. 1

[22] R. n.d. Number of wholesale shipments of RV in the united states from 2000-2016. *Statista*, 2016. 1

[23] F. Oniga and S. Nedevschi. Processing dense stereo data using elevation maps: Road surface, traffic isle, and obstacle detection. *IEEE Trans.Vehicular Technology*, 59(3):1172–1182, 2010. 2

[24] O. M. Parkhi, A. Vedaldi, and A. Zisserman. Deep face recognition. In *Proc. British Machine Vision Conf. (BMVC)*, volume 1, page 6, 2015. 4

[25] S. Ren, K. He, R. Girshick, and J. Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In *Advances in Neural Information Processing Systems (NIPS)*, pages 91–99, 2015. 2

[26] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun. OverFeat: Integrated recognition, localization and detection using convolutional networks. *arXiv preprint arXiv:1312.6229*, 2013. 2

[27] S. Song and M. Chandraker. Joint SFM and detection cues for monocular 3d localization in road scenes. In *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, pages 3734–3742, 2015. 2, 3

[28] A. Vedaldi and K. Lenc. Matconvnet: Convolutional neural networks for matlab. In *Proc. ACM Int. Conf. Multimedia (ICM)*, pages 689–692. ACM, 2015. 8

[29] L. Wang, W. Ouyang, X. Wang, and H. Lu. STCT: Sequentially training convolutional networks for visual tracking. Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR), 2016. 2

[30] Y. Wang and I. H. Witten. Induction of model trees for predicting continuous classes. In *European Conf. Machine Learning*, 1996. 5

[31] R. W. Wolcott and R. M. Eustice. Visual localization within lidar maps for automated urban driving. In *Proc. Int. Conf. Intelligent Robots and Systems (ICIRS)*, pages 176–183. IEEE, 2014. 2

[32] G. Zhu, F. Porikli, and H. Li. Robust visual tracking with deep convolutional neural network based object proposals on PETS. In *Proc. IEEE Conf. Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 26–33, 2016. 2