Infrastructure-free NLoS Obstacle Detection for Autonomous Cars

Felix Naser[1], Igor Gilitschenski[1], Alexander Amini[1], Christina Liao[1], Guy Rosman[2],
Sertac Karaman[3], and Daniela Rus[1]

Abstract— **Current perception systems mostly require direct line of sight to anticipate and ultimately prevent potential collisions at intersections with other road users. We present a fully integrated autonomous system capable of detecting shadows or weak illumination changes on the ground caused by a dynamic obstacle in NLoS scenarios. This additional virtual sensor "ShadowCam" extends the signal range utilized so far by computer-vision ADASs. We show that (1) our algorithm maintains the mean classification accuracy of around 70% even when it doesn't rely on infrastructure – such as AprilTags – as an image registration method. We validate (2) in real-world experiments that our autonomous car driving in night time conditions detects a hidden approaching car earlier with our virtual sensor than with the front facing 2-D LiDAR.**

I. INTRODUCTION

Even though the number of vehicles on the roads is increasing, the number of fatal road accidents is trending downwards in the United States of America (USA) since 1990[1]. This is mostly due to active safety features such as Advanced Driver Assistance Systems (ADAS). Despite this positive trend still around 1.3M fatalities occur due to road accidents every year according to the World Health Organization (WHO)[2]. Specifically dangerous are night time driving scenarios[3] and almost half of the intersection related crashes are caused due to the driver's inadequate surveillance[4]. Better perception systems and increased situational awareness could help to make driving safer.

To deliver on this promise of future mobility solutions with more advanced self-driving capabilities technical approaches both on the hardware and the algorithmic side need to improve. It requires exploring new ways of how each sub-module of an autonomous system's architecture (e.g. perception, planning, and control) could contribute to safer driving in the future.

On the perception side, increasing safety could mean developing more accurate, robust and weather invariant sensors. It could also mean using existing sensors in new ways

[1]Felix Naser, Igor Gilitschenski, Alexander Amini, Christina Liao and Daniela Rus are with the Massachusetts Institute for Technology, Computer Science and Artificial Intelligence Laboratory (CSAIL), Cambridge, MA, USA. {fnaser, igilitschenski, amini, ccliao}@mit.edu and rus@csail.mit.edu
[2]Guy Rosman is with the Toyota Research Institute (TRI), Cambridge, MA, USA. rosman@csail.mit.edu
[3]Sertac Karaman is with the Massachusetts Institute for Technology, Laboratory for Information and Decision Systems (LIDS), Cambridge, MA, USA. sertac@mit.edu

[1]https://goo.gl/8qEpAc
[2]https://goo.gl/vfVgQw
[3]https://goo.gl/CrJnzc
[4]https://goo.gl/KKM2Dn

Fig. 1. Experiment with two cars. The extended ShadowCam [15] algorithm detects a moving obstacle (lights on) behind parked cars from the view point of an autonomous car (lights off) at night time driving conditions even before the LiDAR.

and exploiting new signal ranges which could be used for obstacle detection or early collision warning. This could improve safety by increasing the situational awareness and the perception horizon (i.e. decreasing the number of blind spots) of a human driver or the autonomous car.

Specifically, we aim to detect unexpected dynamic obstacles out of the direct line of sight from the viewpoint of the moving vehicle even at night time driving conditions based on shadows and illumination cues. This would help to detect obstacles behind buildings or parked cars and thus help to prevent collisions (Fig. 1).

Current sensor solutions (e.g. LiDAR, RADAR, Ultrasonic, Cameras, etc.) and algorithms widely used in ADAS applications require a direct line of sight in order to detect and/or classify dynamic obstacles. Some methods can handle partial occlusion of objects but anticipating collisions with unseen obstacles has so far been impossible. The ShadowCam algorithm [15] proposes a solution for non-line-of-sight (NLoS) cases, but the environment needs to be modified by placing AprilTags close to the occlusion.

The results of this paper provide evidence that computer-vision approaches for hidden obstacle detection could ultimately help to make driving safer for pedestrians as well as drivers. Our three key contributions (assuming that the ROI is known, obstacle and vehicle move at slow speeds (ca. 3-5mph) and the obstacle is physically able to cast a shadow or change the illumination) include:

- Extended ShadowCam algorithms run fully integrated on autonomous car
- Extended ShadowCam does not rely on AprilTags and maintains classification accuracy
- Extended ShadowCam runs even at night and can detect approaching cars based on their headlights and shadow (before e.g. a LiDAR can detect it)
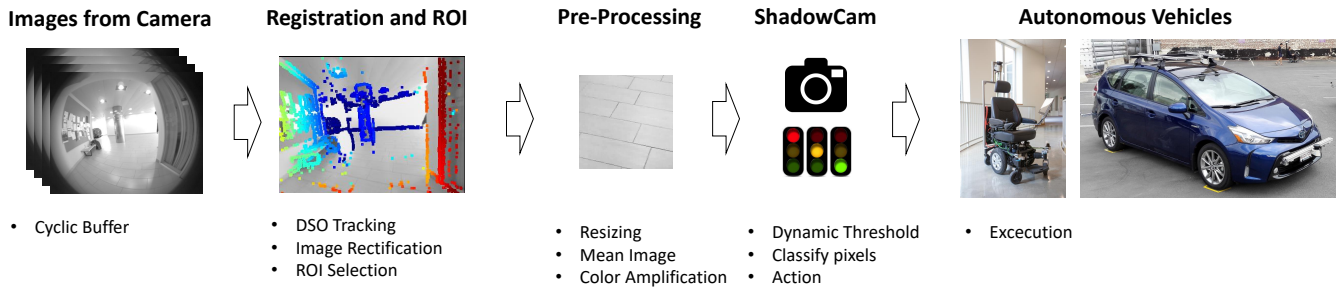
Fig. 2. Overview of the extended ShadowCam Algorithm (based on [15]). We developed an image registration step based on Direct Sparse Odometry (DSO) (instead of AprilTags as proposed in [15]) and integrated the system into an autonomous wheelchair and car.

In the following section (Sec. II) we give an overview of the related works. In Sec. III we introduce our technical approach, specifically how we integrated DSO (Sec. III-C). The experimental setup and data collection procedure are covered in Sec. IV. The results of our technical approach on the dataset are presented in Sec. V. We close the paper with conclusions and future work outlook in Sec. VI.

II. RELATED WORK

This section covers related works and methods previously used to see past or through occlusions (non-line-of-sight (NLoS) problem) and vision based ADAS. Proposed NLoS solutions range from WiFi signals [1], to exploiting specular surfaces [30], [7] and drones [27]. Our presented work does not rely on any infrastructure, hardware or material assumptions. Vision based ADAS research and products are tackling a broad range of problems such as lane-detection-warning (LDW), forward-collision-warning (FDW), traffic sign detection, surround view among others [26]. Most related to our proposed algorithms are the taillight and obstacle detection research areas.

Taillight Detection at Nights. Even though it is statistically more dangerous to drive at night, a survey of the literature suggests that vision based ADAS for night-time driving is less focused on [22], [3]. Instead vision based vehicle detection at day time is covered more broadly [24], [23], [21]. Some taillight detection approaches for vehicles at nights are rule-based [2] and others are learning-based [22]. But both require direct sight of the other vehicle to detect it based on the taillight. Whereas the extended ShadowCam pipeline can detect approaching cars even before they are directly visible.

Pedestrian and Object Detection. Pedestrian or more general object detection systems for ADAS applications undergo a similar trend from rule-based [10] to more learning based approaches [11], [5].

Handling occlusion for ADAS mostly tried to improve the tracking by improving the detectors of the object (e.g. pedestrian or vehicle) [16], [9], [18]. These works assumed partial visibility or a momentary occlusion.

Shadow Processing. So far shadow processing usually focused more on the removal [12], [19], [8]. Only recently it was shown that a 1-D video can be created from a static camera and faint shadows of moving persons [4].

[15] proposes a method to utilize the shadow signal from a moving platform. The video sequences are registered with visual fiducial markers (i.e. AprilTags) on the ground plane. This provides almost perfect image registration. Our approach instead relies on a visual odometry method (i.e. DSO) in order to register the sequences into the same coordinate system. This increases the generalizability of the method since we can run the ShadowCam algorithm on any corner without placing AprilTags markers on the ground plane beforehand. But this also introduces more noise to the system.

III. APPROACH

Our technical approach proposes a solution to the problem of detecting dynamic obstacles out of the direct line of sight from the viewpoint of a moving vehicle based on shadows (Fig. 2). Conceptually we aim to increase safety by increasing the situational awareness of a human driver when ShadowCam is used as an additional ADAS or of the autonomous vehicle when ShadowCam is used as an additional perception module. In this section we highlight the specific challenges of this problem and explain our technical approach to address these. The core extensions of the ShadowCam pipeline are (1) the integration of a visual odometry method for image registration (Sec. III-C) and (2) integration into an autonomous car. This enables the human driver or the autonomous vehicle to avoid potential collisions with dynamic obstacles out of the direct line of sight at day and night time driving conditions.
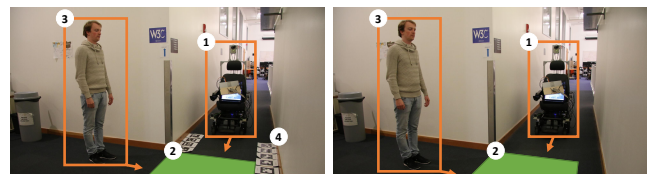

Fig. 3. Problem Setup. We compare the performance of two image registration methods as part of the ShadowCam pipeline: On the left side we place visual fiducial markers (i.e. AprilTags) on the ground plane and on the right we use a visual odometry method (i.e. DSO) for image registration.

Fig. 3 visualizes the problem setup: Number (1) marks the autonomous wheelchair, (2) the known Region of Interest (ROI) where a shadow is expected to be detected and

(3) the dynamic obstacle out of the line of sight. (4) are the visual fiducial markers (i.e. AprilTags) placed on the ground plane. The algorithm from [15] runs on a cyclic buffer and in a pre-processing step projects all images of the buffer to the same viewpoint (Sec. III-A). On these registered image sequences, we run the ShadowCam algorithm to detect dynamic obstacles (Fig. 2). For the registration step we compare two techniques:

- Visual fiducial markers (i.e. AprilTags) placed on the ground plane (Sec. III-B)
- Visual odometry method (i.e. Direct Sparse Odometry (DSO)) to get the rotation matrix and the translation vector between each frame for the projection into the same coordinate system (Sec. III-C)

We use hand annotations for each corner to crop the Region of Interest ROI where we expect to see a shadow (same as in [15]). Other methods of determining the ROI could be map-, place-recognition- or deep-learning-based, but this is not the focus of this work.

The ShadowCam pipeline (Fig. 2) consists of five steps. First, we run a cyclic buffer with the image stream from the camera. Then we run two different image registration methods: Either based on AprilTags (Sec. III-B) or on DSO (Sec. III-C) on this buffer. In Sec. III-A we introduce more details about the image registration process. This step also includes the ROI selection based on the annotations. The output of the second step is a registered buffer (i.e. frame sequence) with ROI selection. During the third step (i.e. pre-processing step) we compute the mean image of the current sequence, resize and amplify the signal. The output of this third step is a frame sequence with the same size for both image registration methods. This allows us to interchange the image registration methods seamlessly. The classification algorithm of the ShadowCam pipeline (Sec. III-D) in the fourth step decides based on the pre-processed image sequence whether it is safe to continue along the path. The vehicle interface in the fifth and last step then executes this decision.

A. Image Registration

In literature, image registration usually refers to the process of transforming multiple images into the same coordinate system. This process can be split into four steps [31]:

- Feature detection (e.g. Oriented FAST and rotated BRIEF (ORB), scale-invariant feature transform (SIFT) or speeded up robust features (SURF))
- Feature matching
- Estimating the homography based on the matched feature points
- Resampling and transformation of the image with an appropriate interpolation technique

These steps lead to (Eq. 1) the homography H which transforms points of two planes (up to a scale-factor s) with 8 Degrees of Freedom (DOF):

$$s \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = H \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \quad (1)$$

This allows to overlay two or more images from the same environment but shot from different angles. We introduce in the following two methods we have chosen for image registration.

B. AprilTags as image registration

We use AprilTags [17], [28] to provide features for sequence registration. AprilTags are a visual fiducial system. The tags can be created from a normal printer, and the open-source AprilTag detection software "computes the precise 3D position, orientation, and identity of the tags relative to the camera"[5]. The open-source implementation is real-time capable. The Alg. 1 summarizes how AprilTags were used in [15].

For all frames in the cyclic buffer we find the maximum set of commonly detected tags (step 3) and compute homographies (step 5) based on the matched points (step 4). This homography then transforms all frames f_i in the buffer to the view point of the first camera frame (transformation from c_i to c_0 in step 6).

Algorithm 1 AprilTag Image Registration

1: $d_0 \leftarrow tagDetection(0)$
2: **for all** i=1; i¡ buffer.length; i++ **do**
3:     $d_i \leftarrow tagDetection(i)$
4:     $m_i \leftarrow findMatchingPoints()$
5:     $H_{c_i}^{c_0} \leftarrow computeHomography()$
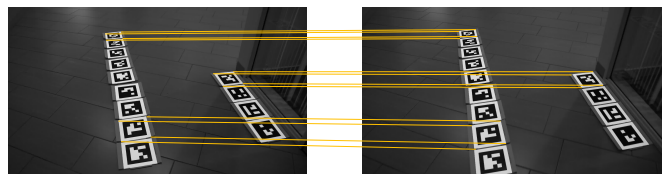6: $f_0 \leftarrow warpPerspective(f_i)$

Fig. 4. AprilTag Matches. Example matches of the AprilTags on the ground plane where the image on the right is closer to the corner. Each tag has a unique ID which allows finding corresponding points fast and easily.

C. DSO for image registration

Many different visual odometry methods have been developed in the past 15 years, with wide ranging of applications in robotics and augmented reality. On a higher level, literature separates this line of work based on the data association design choice [29]: Direct (a) vs. feature based (b) methods. Our choice for DSO is mainly driven by two requirements:

[5]https://april.eecs.umich.edu/software/apriltag

- The code is open-source, works and real-time capable (i.e. ca. 20Hz)
- The visual odometry method should also perform reliably in hallways and areas where only very few textural features exist

Specifically, we looked at the open-source implementations of ORB-SLAM [13] and DSO. But since ORB-SLAM is a feature-based method, it works better in more feature rich environments. We run our experiments primarily in hallways without many textural features. In theory DSO performs more reliably in this setting. DSO is a sparse and direct method for monocular visual odometry. It "jointly optimizes the full likelihood for all involved model parameters, including camera poses, camera intrinsics, and geometry parameters (inverse depth values)" [6]. These initial tests confirmed that DSO performs better in our experiment settings.

After this initial evaluation, we moved forward with DSO. We adapted and modified the open-source code so that it integrates seamlessly the ShadowCam pre-processing pipeline.

The open-source implementation of DSO[6] computes the pose for each frame (M_w^c), which is composed of the rotation matrix R and the translation vector t. In the following section, we describe how we obtain the homography H mathematically from R and t. The homography is proportional to the information given by the *planar surface* equation, the rotation matrix R and the translation vector t between two image frames

$$H \propto R - tn^T \tag{2}$$

where n designates the normal of the local planar approximation of the scene [20]. Symbols used in this section are described in Table I.

Algorithm 2 gives an overview of how the following equations are connected to get H from R and t for each frame. We obtain R and t for the first frame in the buffer and register all following frames with respect to the first frame (in Alg. 2 denoted as c_2). This essentially means that all frames are projected into the same coordinate system.

We annotate three points on the ground plane and in the world frame w. This results in reasonable transformations for most pixels on the ground plane. This is important because later in the pipeline, we want to classify shadows close to a corner on this plane.

After we obtain R and t of frame f_i we can transform the points on the plane and in the world frame w to the camera frame c_1. M_w^c in homogeneous form[7] transforms points from the world frame (denoted as w) into the camera frame (denoted as c):

$$\begin{bmatrix} X_c \\ Y_c \\ Z_c \\ 1 \end{bmatrix} = M_w^c \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} = \begin{bmatrix} R_w^c & t_w^c \\ 0_{1\times3} & 1 \end{bmatrix} \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} \tag{3}$$

Algorithm 2 DSO Image Registration

1: $planePoints_w \leftarrow parametersFromFile()$
2: $R_{c_2} \leftarrow getRotationMatrix(0)$ ▷ Rotation matrix of first frame in cyclic buffer
3: $t_{c_2} \leftarrow getTranslationVector(0)$ ▷ Translation vector of first frame in cyclic buffer
4: **for all** i=1; i¡ buffer.length; i++ **do**
5:      $R_{c_1} \leftarrow getRotationMatrix(i)$
6:      $t_{c_1} \leftarrow getTranslationVector(i)$
7:      $planePoints_{c_1} \leftarrow$ Eq. 3 ▷ Transformation of world plane points to c_1
8:      $R_{c_1}^{c_2} \leftarrow$ Eq. 6 ▷ Obtaining rotation matrix from c_1 to c_2
9:      $t_{c_1}^{c_2} \leftarrow$ Eq. 7 ▷ Obtaining translation vector from c_1 to c_2
10:      $n_{c_1} \leftarrow computeNormal(planePoints_{c_1})$
11:      $d_{c_1} \leftarrow computeDistance()$
12:      $H_{c_1}^{c_2} \leftarrow$ Eq. 8 ▷ Calculating homography matrix
13:      $f_{c_2} \leftarrow warpPerspective(f_{c_1})$

Given K the camera's intrinsic matrix and M_w^c the camera's pose we can obtain the image points directly from world points in the following way:

$$s \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \end{bmatrix} \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} \tag{4}$$

With both positions of the camera ($M_w^{c_1}$ and $M_w^{c_2}$, where c_2 is the camera frame of the first image in the cyclic buffer) we can find the transformation for a 3D point from camera frame c_1 to c_2:

$$\begin{aligned} M_{c_1}^{c_2} &= M_w^{c_2} \cdot (M_w^{c_1})^{-1} \\ &= \begin{bmatrix} R_w^{c_2} & t_w^{c_2} \\ 0_{3\times1} & 1 \end{bmatrix} \cdot \begin{bmatrix} (R_w^{c_1})^T & -(R_w^{c_1})^T \cdot t_w^{c_1} \\ 0_{1\times3} & 1 \end{bmatrix} \end{aligned} \tag{5}$$

This allows us to specify the rotation matrix R

$$R_{c_1}^{c_2} = R_w^{c_2} \cdot (R_w^{c_2})^T \tag{6}$$

and the translation vector t between two frames

$$t_{c_1}^{c_2} = R_w^{c_2} \cdot \left(-(R_w^{c_1})^T \cdot t_w^{c_1} \right) + t_w^{c_2} \tag{7}$$

With the distance d as the dot product between the plane normal and a point on the plane, this leads to the homography H from c_1 to c_2

$$H_{c_1}^{c_2} = R_{c_1}^{c_2} - \frac{t_{c_1}^{c_2} \cdot (n_{c_1})^T}{d_{c_1}} \tag{8}$$

which is the same as Eq. 2 including scaling.

D. ShadowCam Classifier

The classifier is the same as proposed in [15]. The core parts of the classifier amplify a weak signal and distinguish sequences into "dynamic" or "static" depending on whether a

---

TABLE I

SYMBOL TABLE. DESCRIPTION OF VARIABLES USED IN THIS SECTION.

Symbol	Description
M_c^w	Camera pose, transformation from camera c to world w frame (4x4 matrix)
R_c^w	Rotation matrix, rotation from camera c to world w frame (3x3 matrix)
t_c^w	Translation vector, translation from camera c to world w frame (3x1 matrix)
$H_{c_1}^{c_2}$	Homography matrix, projection from camera c_1 to c_2 frame (3x3 matrix)
K	Camera intrinsics (3x3 matrix)
n_{c_1}	Plane normal in camera frame c_1 (3x1 matrix)
d_{c_1}	Distance between camera c_1 and plane (skalar)

moving obstacle was around the corner by using a threshold based on mean and standard deviation.

In summary, in this section we present our technical approach to tackle the problem of detecting moving obstacles out of the direct line of sight from the view point of the vehicle based on shadows. We incorporate two image registration methods in the same pipeline. During pre-processing we amplify the sometimes-weak shadow signal. The decision of whether it is safe to move ahead is based on a pixel sum per sequence and a threshold.

IV. EXPERIMENTAL SETUP

This section gives an overview of how and under which conditions we collected the dataset to evaluate the technical approach from Sec. III. Sec. V then presents the performance of our technical approach on the dataset we present in this chapter. In general, we want to compare the classification accuracy between AprilTags and no AprilTags and between "dynamic" and "static" sequences. Thus, we are interested in collecting data in the real-world under four main circumstances (Fig. 5):

- AprilTags with dynamic obstacle around corner (i.e. "dynamic" sequence)
- AprilTags without dynamic obstacle around corner (i.e. "static" sequence)
- No AprilTags with dynamic obstacle around corner (i.e. "dynamic" sequence)
- No AprilTags without dynamic obstacle around corner (i.e. "static" sequence)

In addition to the comparison of AprilTags vs. no AprilTags we want to show how the extended ShadowCam pipeline performs on an autonomous car at night time driving conditions (Sec. IV-B).

A. Cameras and Corners

We created a real-world dataset with 4 different cameras. With the Canon EOS 70D and the EFS $17 - 58$ mm lens (single-lens reflex (SLR)) camera, we collected around 1 hour of data resulting in ca. $85,000$ images and 7.4 GB in total. With the camera uEye UI-3241LE-M-GL (monochrome, global shutter CMOS) from IDS[8] we

[8] https://www.ids-imaging.us/home.html

collected around $42,000$ images at around 20Hz resulting in ca. 73.4 GB in total.

As in [15] the camera is moving in a range of 1 to 3 meters back and forth at around 3mph, whereas the person behind the corner moves randomly in a similar range and pace.

We collected data to cover a broad range of nuisance factors, such as size of the object, speed of the movement, lighting, reflection properties of the floor, color of the floor, ego motion, among others.

Fig. 5. Real-world corner examples. On the left side images from videos recorded with the Canon and AprilTags. On the right side images of the same corners from videos recorded with the IDS uEye and DSO. The dataset in total consists out of 7 corners.

B. Autonomous Vehicles

The autonomous systems – wheelchair and car (based on [14]) – operate in a given map with a pre-defined path. The localization approach is based on laser scan matching (AMCL [25]). The re-planning in case of a moving obstacle is using an RRT* variant (rapidly exploring random tree). Path following is done with a pure pursuit controller implementation (Fig. 6). The integration of the ShadowCam pipeline is not yet perfect but it does showcase its initial functionality.

To enable DSO we upgraded the camera on the wheelchair to a global shutter camera which can run up to 60 fps. For the experiments we run it at 20 fps. In a distributed setup where one laptop runs the autonomous software and the other laptop runs the ShadowCam algorithm we can output classication results at around 20 Hz.

V. RESULTS

We quantitatively analyze the classification accuracy, real-time capability of the algorithm and demonstrate the use of ShadowCam integrated into an autonomous car and wheelchair. Specifically, we are evaluating the performance
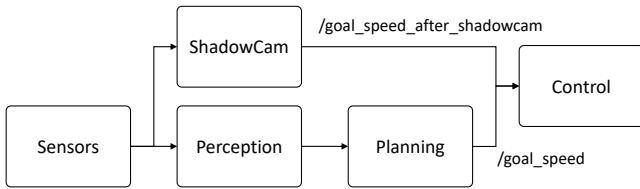
Fig. 6. Autonomous System Architecture. This architecture only demonstrates the concept of the ShadowCam pipeline where the control module actuates the car based on $min($/goal_speed_after_shadowcam,/goal_speed$)$. In a next iteration the planning module should directly incorporate the ShadowCam signal and make the stop/go decision.

of two image registration methods (AprilTags from Sec. III-B and DSO from Sec. III-C) and compare the classification accuracy of "dynamic" and "static" sequences. The success metric is as follows e.g.: When the ShadowCam pipeline classifies 7 out of 10 "static" sequences as "static" the classification accuracy would be 70%.

Boxplots, histograms and Receiver-Operating-Characteristic (ROC) analysis visualize the performance of the extended ShadowCam algorithm on the respective datasets in Fig. 7, 8, 9 comparing AT versus DSO.

A. Wheelchair: Comparison AprilTags and DSO

We run the ShadowCam algorithm on 7 corners using AT (= AprilTags) (Sec. III-B) and DSO (= Direct Sparse Odometry) (Sec. III-B) as image registration methods. Our experiments give further evidence that it is possible to detect moving obstacles from a moving viewpoint at indoor corners where it is physically possible for a dynamic obstacle to cast a shadow at relatively slow speeds (e.g. 3-5mph). Fig. 7 indicates the classification accuracy for each data collection mode. Importantly we can observe that for both classes "static" and "dynamic" the accuracy is well above *random* 50%. Additionally, even when we remove AprilTags and rely instead on DSO as the image registration method, we can maintain a classification accuracy of around 70%. Overall, we can also observe that the classification accuracy for "static" sequences is higher than it is for "dynamic" shadows.

As introduced in Sec. IV the dataset for the AprilTag case is around 60 mins and around 4000 sequences in size, while for the DSO case it is around 40 mins and 1500 sequences in size (where for both cases each sequence consists out of 10 frames). This adds up to around 100 mins and 5500 sequences of real-world experiment data. With a mean classification accuracy of around 70% for both image registration methods, this means that ShadowCam classifies 3850 sequences or 70 mins correctly into the categories "dynamic" or "static" depending on whether a dynamic obstacle was moving behind the corner. This helps to prevent a potential collision with a "dynamic" obstacle out of the direct line of sight. Since we aim for an algorithm parametrization which allows a smooth driving experience, ShadowCam only outputs a "stop" signal when the movement behind the corner is relatively strong. Thus, the classification accuracy for both image registration methods is higher for "static" sequences.
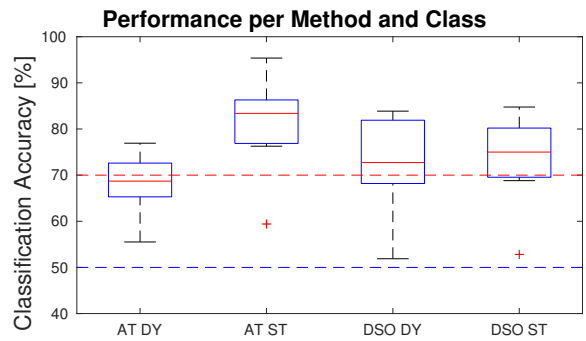

Fig. 7. AprilTags and DSO classification accuracy per class (ST = static and DY = dynamic).
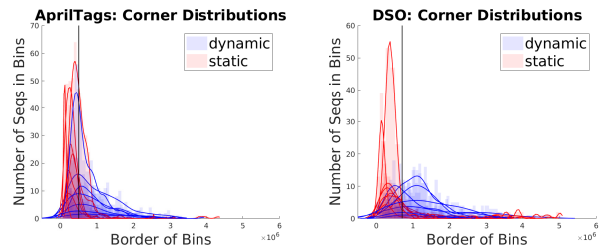

Fig. 8. Signal Distributions. Histograms of the sum of the sequences for various corners with AprilTags on the left and without AprilTags and DSO as the image registration method on the right. The distributions of "dynamic" and "static" sequences explain the mean classification accuracy of around 70% when the threshold is set as the black vertical line indicates.
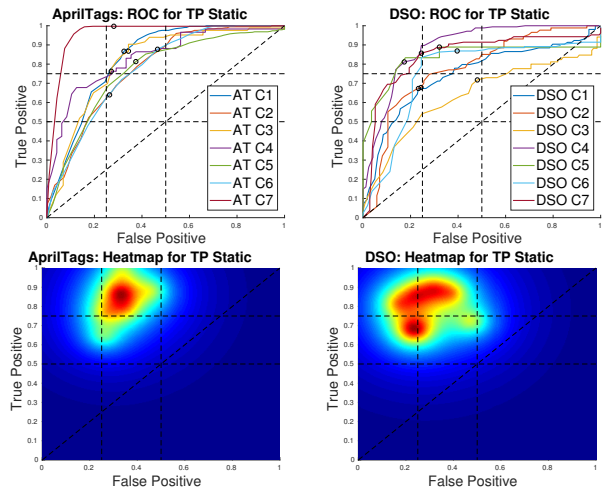

Fig. 9. Receiver-Operating-Characteristic (ROC). Curves of data collected with AprilTags on the left and without AprilTags but instead with DSO. TP stands for true positive. If the heatmaps would have their center in the top left corner the classification accuracy would be 100%.

The plots (Fig. 7, 8, 9) indicate coherent trends. DSO is weaker on ST sequences and stronger on DY than AT. This is for example reflected in the heatmap (Fig. 9) with the center of AT being higher and more to the right, where higher means a higher true positive rate for ST sequences and further to the right a higher false positive rate.

Fig. 10. Garage Experiment with DSO as the image registration method. The extended ShadowCam algorithm is able to detect the approaching car even before the front-facing LiDAR is able to detect it. This is enabled by the signal amplification relying on the registered frames within a sequence. The faint white signals on the right are visualizing how the ShadowCam can detect the approaching car based on weak illumination changes.

B. Car: Garage Experiments

Besides the safety and regulatory reasons, running experiments in the garage allows us to test performance close to night time driving conditions. The dataset where we compare AT vs. DSO performance (Sec. V-A) covers indoor corners at day times.

The lights of the autonomous vehicle are turned off during the experiments since we would have to incorporate an ego motion estimation. The headlights at night time driving could cause the ShadowCam to detect the ego motion instead of detecting an unseen dynamic obstacle approaching from the right during a left turn.

In Fig. 10 we compare two time steps of the experiment[9]:

- $t_1 = 52.15$ sec (first row): ShadowCam detects approaching car from the right
- $t_2 = 52.87$ sec (second row): LiDAR detects approaching car from the right

The extended ShadowCam pipeline is able to detect an approaching car earlier than a LiDAR. The ShadowCam pipeline runs at 20Hz. This experiment depends on an accurate annotation of the ROI and ground plane. We also tuned the threshold specifically for the garage light conditions.

Furthermore, we ran this experiment three times non-automated with two cars and two drivers to evaluate the classification accuracy of our algorithm in this setup. 94 sequences were annotated as "static" and 31 sequences as "dynamic" (Fig. 11). In comparison to the wheelchair experiments (Sec. V-A) the accuracy is relatively high since we specifically tuned the threshold for the light conditions in the garage.

VI. DISCUSSION AND CONCLUSION

One conceptual way to improve safety is to increase situational awareness which could benefit a human driver as well an autonomous vehicle. We give further evidence that
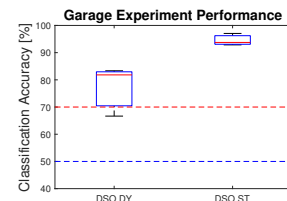

Fig. 11. In the garage experiment setup our algorithm achieves a combined mean-classification accuracy of 85.91%.

this can be achieved not only by developing new sensors but also by exploring under-utilized signal ranges, in our case visual shadow signals where the focus usually lies on removing them.

We extend the ShadowCam pipeline from [15]. We can show a classification accuracy even without AprilTags for the image registration step around 70%. Additionally, we present real-world experiments with an autonomous car and the ShadowCam pipeline at night time driving conditions. This showcases that even before traditional ADAS perception systems (e.g. LiDAR) can detect a dynamic obstacle, our proposed solution can help to prevent collisions.

In the future, we want to explore more data and deep learning driven approaches to achieve higher classification accuracy. We plan to integrate an automated ROI detection and ego-motion estimation into the processing pipeline.

ACKNOWLEDGEMENTS

[9] https://youtu.be/NqFEd-9J1lE

REFERENCES

[1] F. Adib and D. Katabi. See Through Walls with WiFi! In *Proceedings of the ACM SIGCOMM Conference*, 2013.

[2] A. Almagambetov, S. Velipasalar, and M. Casares. Robust and computationally lightweight autonomous tracking of vehicle taillights and signal detection by embedded smart cameras. *IEEE Transactions on Industrial Electronics*, 62(6):3732–3741, June 2015.

[3] N. Boonsim and S. Prakoonwit. An algorithm for accurate taillight detection at night. *International Journal of Computer Applications*, 100(12), August 2014.

[4] K. L. Bouman, V. Ye, A. B. Yedidia, F. Durand, G. W. Wornell, A. Torralba, and W. T. Freeman. Turning corners into cameras: Principles and methods. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[5] Z. Cai, Q. Fan, R. S. Feris, and N. Vasconcelos. A unified multiscale deep convolutional neural network for fast object detection. In *European Conference on Computer Vision*, pages 354–370. Springer, 2016.

[6] J. Engel, V. Koltun, and D. Cremers. Direct sparse odometry. *CoRR*, abs/1607.02565, 2016.

[7] R. Fergus, A. Torralba, and W. T. Freeman. Random lens imaging. *CSAIL Technical Report*, 2006.

[8] G. D. Finlayson, M. S. Drew, and C. Lu. Entropy Minimization for Shadow Removal. *International Journal of Computer Vision*, 85(1):35–57, 2009.

[9] T. Frank, M. Haag, H. Kollnig, and H.-H. Nagel. Tracking of Occluded Vehicles in Traffic Scenes. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 1996.

[10] D. M. Gavrila and M. Enzweiler. Monocular pedestrian detection: Survey and experiments. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31:2179–2195, 10 2008.

[11] D. Geronimo, A. M. Lopez, A. D. Sappa, and T. Graf. Survey of pedestrian detection for advanced driver assistance systems. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(7):1239–1258, July 2010.

[12] S. H. Khan, M. Bennamoun, F. Sohel, and R. Togneri. Automatic Shadow Detection and Removal from a Single Image. *Transactions on Pattern Analysis and Machine Intelligence*, 38(3):431–446, 2016.

[13] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardos. Orb-slam: A versatile and accurate monocular slam system. *IEEE Transactions on Robotics*, 31(5):1147–1163, Oct 2015.

[14] F. Naser, D. Dorhout, S. Proulx, S. D. Pendleton, H. Andersen, W. Schwarting, L. Paull, J. Alonso-Mora, M. H. Ang, S. Karaman, R. Tedrake, J. Leonard, and D. Rus. A Parallel Autonomy Research Platform. In *Proceedings of the Intelligent Vehicles Symposium (IV)*, 2017.

[15] F. Naser, I. Gilitschenski, G. Rosman, A. Amini, F. Durand, A. Torralba, G. W. Wornell, W. T. Freeman, S. Karaman, and D. Rus. Shadowcam: Real-time detection of moving obstacles behind A corner for autonomous vehicles. In *Proceedings of International Conference on Intelligent Transportation Systems (ITSC)*, 2018.

[16] E. Ohn-Bar and M. M. Trivedi. Learning to Detect Vehicles by Clustering Appearance Patterns. *Transactions on Intelligent Transportation Systems*, 16(5):2511–2521, 2015.

[17] E. Olson. AprilTag: A Robust and Flexible Visual Fiducial System. In *Proceedings of the International Conference on Robotics and Automation (ICRA)*, 2011.

[18] W. Ouyang and X. Wang. A Discriminative Deep Model for Pedestrian Detection with Occlusion Handling. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[19] R. Ramakrishnan, J. Nieto, and S. Scheding. Shadow compensation for outdoor perception. In *Proceedings of the International Conference on Robotics and Automation (ICRA)*, 2015.

[20] G. Rosman, S. Shem-Tov, D. Bitton, T. Nir, G. Adiv, R. Kimmel, A. Feuer, and A. M. Bruckstein. Over-parameterized optical flow using a stereoscopic constraint. In A. M. Bruckstein, B. M. ter Haar Romeny, A. M. Bronstein, and M. M. Bronstein, editors, *Scale Space and Variational Methods in Computer Vision*, pages 761–772, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg.

[21] R. K. Satzoda and M. M. Trivedi. Efficient lane and vehicle detection with integrated synergies (elvis). In *2014 IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 708–713, June 2014.

[22] R. K. Satzoda and M. M. Trivedi. Looking at vehicles in the night: Detection and dynamics of rear lights. *IEEE Transactions on Intelligent Transportation Systems*, 99:1–11, 2016.

[23] R. K. Satzoda and M. M. Trivedi. Multipart vehicle detection using symmetry-derived analysis and active learning. *IEEE Transactions on Intelligent Transportation Systems*, 17(4):926–937, April 2016.

[24] S. Sivaraman and M. M. Trivedi. Looking at vehicles on the road: A survey of vision-based vehicle detection, tracking, and behavior analysis. *IEEE Transactions on Intelligent Transportation Systems*, 14(4):1773–1795, Dec 2013.

[25] S. Thrun, D. Fox, W. Burgard, and F. Dellaert. Robust monte carlo localization for mobile robots. *Artificial intelligence*, 128(1-2):99–141, 2001.

[26] G. Velez and O. Otaegui. Embedding vision-based advanced driver assistance systems: a survey. *IET Intelligent Transport Systems*, 2016.

[27] A. Wallar, B. Araki, R. Chang, J. Alonso-Mora, and D. Rus. Foresight: Remote Sensing for Autonomous Vehicles Using a Small Unmanned Aerial Vehicle. In *Proceedings of the Conference on Field and Service Robotics (FSR)*, 2018.

[28] J. Wang and E. Olson. AprilTag 2: Efficient and Robust Fiducial Detection. In *Proceedings of the International Conference on Intelligent Robots and Systems (IROS)*, 2016.

[29] G. Younes, D. C. Asmar, and E. A. Shammas. A survey on non-filter-based monocular visual SLAM systems. *CoRR*, abs/1607.00470, 2016.

[30] Z. Zhang, P. Isola, and E. H. Adelson. Sparkle vision: Seeing the world through random specular microfacets. *CoRR*, abs/1412.7884, 2014.

[31] B. Zitova and J. Flusser. Image registration methods: a survey. *Image and vision computing*, 21(11):977–1000, 2003.