# J-MOD$^2$: Joint Monocular Obstacle Detection and Depth Estimation

Michele Mancini[1], Gabriele Costante[1], Paolo Valigi[1] and Thomas A. Ciarfuglia[1]

***Abstract*— In this work, we give a new twist to monocular obstacle detection. Most of the existing approaches either rely on Visual SLAM systems or on depth estimation models to build 3D maps and detect obstacles. Despite their success, these methods are not specifically devised for monocular obstacle detection. In particular, they are not robust to appearance and camera intrinsics changes or texture-less scenarios. To overcome these limitations, we propose an end-to-end deep architecture that jointly learns to detect obstacle and estimate their depth. The multi task nature of this strategy strengthen both the obstacle detection task with more reliable bounding boxes and range measures and the depth estimation one with robustness to scenario changes. We call this architecture J-MOD$^2$. We prove the effectiveness of our approach with experiments on sequences with different appearance and focal lengths. Furthermore, we show its benefits on a set of simulated navigation experiments where a MAV explores an unknown scenario and plans safe trajectories by using our detection model.**

## I. Introduction

Obstacle avoidance has been deeply studied in robotics due to its crucial role for vehicle navigation. Recently, the demand for faster and more precise Micro Aerial Vehicle (MAV) platforms has put even more attention on it. To safely execute aggressive maneuvers in unknown scenarios, the MAVs need a robust obstacle detection procedure.

Most fruitful approaches rely on range sensors, such as laser-scanner [1], [2], stereo cameras [3], [4] or RGB-D cameras [5], [6] to build 3D maps and compute obstacle-free trajectories. However, their use results in an increased weight and power consumption, which is unfeasible for small MAVs. Furthermore, their sensing range is either limited by device characteristics (RGB-D and lasers) or by camera baselines (stereo cameras).

Monocular Visual SLAM (VSLAM) approaches address the above limitations by exploiting single camera pose estimation and 3D map reconstruction [7], [8], [9], [10], [11]. Nevertheless, these strategies have multiple drawbacks when fast obstacle avoidance is required: the absolute scale is not observable (which easily results in wrong obstacle distance estimations); they fail to compute reliable 3D maps on low-textured environments; the 3D map updates are slow with respect to real-time requirements of fast maneuvers.

A step toward more robust obstacle detection has been made by monocular depth estimation methods based on Convolutional Neural Networks (CNNs) [12], [13], [14].

*We gratefully acknowledge the support of NVIDIA Corporation with the donation of the Titan X GPU used for this research.

[1]All the authors are with the Department of Engineering, University of Perugia, via Duranti 93, Perugia Italy

{thomas.ciarfuglia, paolo.valigi, gabriele.costante, michele.mancini}@unipg.it

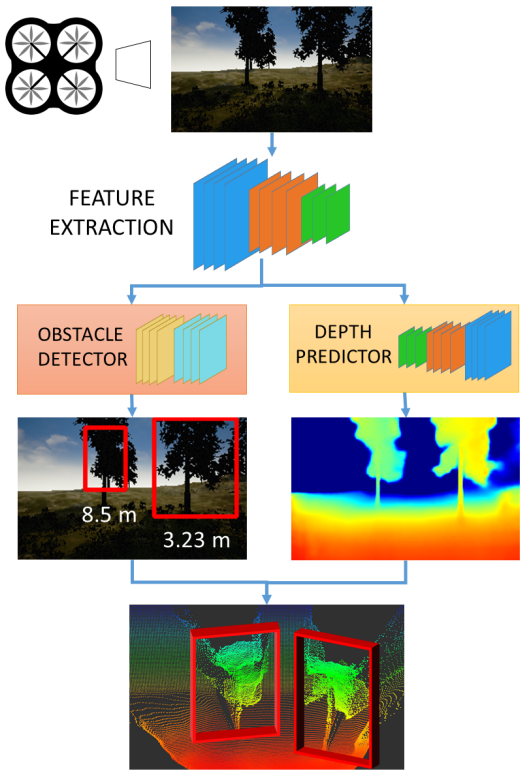



Fig. 1: Overview of the proposed system: the architecture is composed by two networks that perform different, but connected tasks: obstacle detection and pixel-wise depth estimation. The two task are jointly learned and the feature extraction layers are in common. Thus, the resulting model has increased accuracy in depth prediction because of the semantic information received from the detector. On the other hand, the detector learns a better representation of obstacles through depth estimation.

Compared to standard VSLAM strategies, these works train CNN-based model to quickly compute depth maps from single image, which allows for fast trajectory replanning. However, these depth models are biased with respect to appearance domains and camera intrinsics. Furthermore, the CNN architectures so far proposed address the more general task of pixel-wise depth prediction and are not specifically devised for obstacle detection.

Driven by the previous considerations, in this work we propose a novel CNN architecture for monocular obstacle detection that is both fast and robust to focal length and appearance changes. Our model provides obstacle bounding boxes together with their depth to enable the navigation system to compute their 3D position and, consequently, avoid

them. We achieve this by introducing a multi-task CNN architecture that jointly learns to predict full depth maps and obstacle bounding boxes. The combination of these two tasks gives them mutual advantages: the depth prediction branch is informed with object structures, which result in more robust estimations. On the other hand, the obstacle detection model exploits the depth information to predict obstacle distance and bounding boxes more precisely.

We demonstrate the effectiveness of our approach in both publicly available and brand new sequences. In these experiments, we prove the robustness of the learned models in test scenarios that differ from the training ones with respect to focal length and appearance. Furthermore, we set up a full navigation system in a simulated environment with a MAV that detects obstacle and computes free trajectories as it explores the scene.

## II. Related Work

The most straight-forward approaches to obstacle detection involve RGB-D or stereo cameras. These sensors can be used to get a 3D map of the environment, when coupled with a reliable robot state estimation system [15], [6]. Unfortunately, these sensors suffer from limited range. For example, a Microsoft Kinect RGB-D sensor has a declared maximum range of 5 meters, while in stereo rigs with baselines compatible with MAV's sizes, performances start to degrade on distances larger than 3 meters [16]. Range can be slightly increased employing larger baselines, but on MAVs this is usually difficult, due to payload and encumbrance constraints. Some authors explored push-broom stereo systems on fixed-wing, high speed MAVs [17], [18]. However, these approaches use higher baselines, thanks to the large wing span, and, thus, are not easily adaptable to small rotary wing MAVs. In addition, while short-range estimations still allows safe collision avoidance, it sets an upper bound to the robot's maximum operative speed. For all these reasons we chose to use monocular camera systems, that are cheap, easy to use, and light-weight. Nonetheless, our method can detect and localize obstacles up to 20 meters and compute dense depth maps up to 40 meters with a minor payload and space consumption.

Monocular obstacle detection can be achieved by dense 3D map reconstruction via SLAM or Structure from Motion (SFM) based procedures [7], [10], [19], [20]. While these systems are handier to deploy, they carry along additional challenges. In particular, geometric based 3D reconstruction algorithms rely on the triangulation of consecutive frames. Despite the impressive results achieved by State-of-the-Art (SotA) approaches, their accuracy drops during high-speed motion, as dense alignment becomes more challenging. In addition, with standard geometric monocular systems it is not possible to recover the absolute scale of the objects. This prevents them to accurately estimate obstacle distances. For this reason, most approaches exploit optical information to detect proximity of obstacles from camera, or, similarly, detect traversable space [21], [22], [23], [24].

To overcome the limitations of geometric monocular algorithms, deep learning-based solutions have been proposed. These models produce a dense 3D representation of the environment from a single image, exploiting the knowledge acquired through training on large labeled datasets, both real-world and synthetic [25], [12], [26], [13]. A few of these methods have been recently tested in obstacle detection and autonomous flight applications. In [27], the authors fine-tune on a self-collected dataset the global coarse depth estimation model proposed by [25]. Then, they compute MAV's control signals during test flights on the basis of the estimated depth. In [14] the authors exploit depth and normals estimations of a deep model presented in [12] as an intermediate step to train an visual reactive obstacle avoidance system. Differently from [14] and [27], we propose to learn a more robotic-focused model that jointly learns depth and obstacle representations to strengthen the obstacle detection task.

While existent depth estimators achieve impressive results on several computer vision benchmarks, such as NYU-V2 [28] or KITTI [29], they have been trained and evaluated on pixel-wise losses and metrics (*e.g.*, Log RMSE) that do not take into account image semantics. In [30], the authors measure per-semantic-class errors of SotA methods on NYU-V2, showing how all methods significantly perform better on the "ground" class than on other classes (e.g "furniture", "props"). In typical outdoor scenarios, the majority of pixels in a image belong either to "ground" or "sky" classes. Our intuition is that current depth estimators tend to better optimize their predictions on these classes, as they tend to have more regular texture and geometric structures. On the contrary, in robotic applications we want to train detection models to be as accurate as possible when estimating obstacle distances.

For all these reasons, we propose a novel solution to the problem by jointly training a multi-task model for depth estimation and obstacle detection. While each task's output comes from independent branches of the network, feature extraction from their common RGB input is shared for both targets. This choice improves both depth and detection estimations compared to single task models, as shown in the experiments.

For the obstacle detection task, inspired by [31], we regress bounding boxes anchor points and dimensions, but with a slightly different implementation. In particular, we remove the fully connected layers, maintaining a fully convolutional architecture. We favor this strategy over other object detection approaches because of its high computational efficiency, as it allows multiple bounding box predictions with a single forward pass. In addition, we also let the obstacle detector regress the average depth and the corresponding estimate variance of the detected obstacles.

Depth estimation is devised following the architecture of [13], improved by taking into account the obstacle detection branch. In particular, we correct the depth predictions by using the mean depth estimates computed by the obstacle detection branch to achieve robustness with respect to appearance changes. We prove the benefits of this strategy by

validating the model in test sequences with different focal length and scene appearance.

## III. Network Overview

Our proposed network is depicted in Figure 2. Given an $256 \times 160$ RGB input, features are extracted with a fine-tuned version of the VGG19 network pruned of its fully connected layers [32]. VGG19 weights are initialized on the image classification task on the ImageNet dataset. Features are then fed to two, task-dependent branches: a depth prediction branch and a obstacle detector branch. The former is composed by 4 upconvolution layers and a final convolution layer which outputs the predicted depth at original input resolution. This branch, plus the VGG19 feature extractor, is equivalent to the fully convolutional network proposed in [13]. We optimize depth prediction on the following loss:

$$\begin{aligned} L_{depth} = & \frac{1}{n}\sum_i d_i^2 - \frac{1}{2n^2}(\sum_i d_i)^2 \\ & + \frac{1}{n}\sum_i [\nabla_x D_i + \nabla_y D_i] \cdot N_i^* \end{aligned} \tag{1}$$

where $d_i = \log D_i - \log D_i^*$, $D_i$ and $D_i^*$ are respectively the predicted and ground truth depths at pixel $i$, $N_i^*$ is the ground truth 3D surface normal, and $\nabla_x D_i$, $\nabla_y D_i$ are the horizontal and vertical predicted depth gradients. While the first two terms correspond to the scale invariant log RMSE loss introduced in [25], the third term enforces orthogonality between predicted gradients and ground truth normals, aiming at preserving geometrical coherence. With respect to the loss proposed in [12], that introduced a L2 penalty on gradients to the scale invariant loss, our loss performs comparably in preliminary tests.

The obstacle detection branch is composed by 9 convolutional layer with Glorot initialization. The detection methodology is similar to the one presented in [31]: the input image is divided into a $8 \times 5$ grid of square-shaped cells of size $32 \times 32$ pixels. For each cell, we train a detector to estimate:

- The $(x, y)$ coordinates of the bounding box center
- The bounding box width $w$ and height $h$
- A confidence score $C$
- The average distance of the detected obstacle from the camera $m$ and the variance of its depth distribution $v$

The resulting output has a $40 \times 7$ shape. At test time, we consider only predictions with a confidence score over a certain threshold. We train the detector on the following loss:

$$\begin{aligned} L_{det} = \quad & \lambda_{coord} \sum_{i=0}^{N} [(x_i - x_i^*)^2 + (y_i - y_i^*)^2] \\ & + \lambda_{coord} \sum_{i=0}^{N} [(w_i - w_i^*)^2 + (h_i - h_i^*)^2] \\ & + \lambda_{obj} \sum_{i=0}^{N} (C_i - C_i^*)^2 + \lambda_{noobj} \sum_{i=0}^{N} (C_i - C_i^*)^2 \\ & + \lambda_{mean} \sum_{i=0}^{N} (m_i - m_i^*)^2 + \lambda_{var} \sum_{i=0}^{N} (v_i - v_i^*)^2 \end{aligned} \tag{2}$$

where we set $\lambda_{coord} = 0.25$, $\lambda_{obj} = 5.0$, $\lambda_{noobj} = 0.05$, $\lambda_{mean} = 1.5$, $\lambda_{var} = 1.25$. Our network is trained simultaneously on both tasks. Gradients computed by each loss are backpropagated through their respective branches and the shared VGG19 multi-task feature extractor.

### A. Exploiting detection to correct global scale estimations

The absolute scale of a depth estimation is not observable from a single image. However, learning-based depth estimators are able to give an accurate guess of the scale under certain conditions. While training, these models implicitly learn domain-specific object proportions and appearances. This helps the estimation process in giving depth maps with correct absolute scale. As the relations between object proportions and global scale in the image strongly depend on camera focal length, at test time the absolute scale estimation are strongly biased towards the training set domain and its intrinsics. For these reasons, when object proportions and/or camera parameters change from training to test, scale estimates quickly degrade. Nonetheless, if object proportions stay roughly the same and only camera intrinsics are altered at test time, it is possible to employ some recovery strategy. If the size of a given object is known, we can analytically compute its distance from the camera and recover the global scale for the whole depth map. For this reason, we suppose that the obstacle detection branch can help recovering the global scale when intrinsics change. We hypothesize that, while learning to regress obstacles bounding boxes, a detector model implicitly learns sizes and proportions of objects belonging to the training domain. We can then evaluate estimated obstacle distances from the detection branch and use them as a tool to correct dense depth estimations. Let $m_j$ be the average distance of the obstacle $j$ computed by the detector, $\hat{D}_j$ the average depth estimation within the $j$-th obstacle bounding box, $n_o$ the number of estimated obstacles, then we compute the correction factor $k$ as:

$$k = \frac{\frac{1}{n_o}\sum_j^{n_o} m_j}{\frac{1}{n_o}\sum_j^{n_o} \hat{D}_j} \tag{3}$$

Finally, we calculate the corrected depth at each pixel $i$ as $\tilde{D}_i = kD_i$. To validate our hypothesis, in Section IV-D we

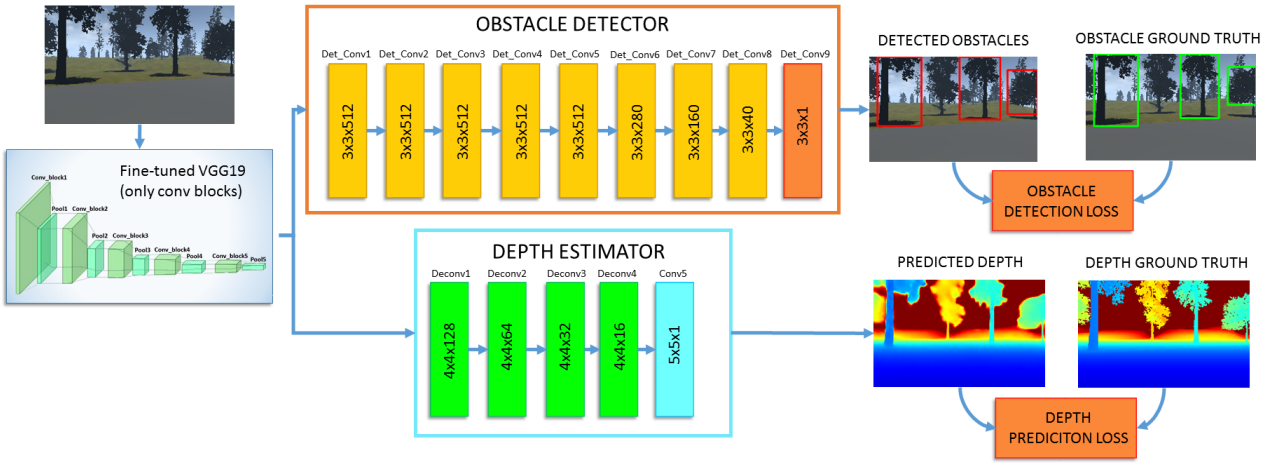



Fig. 2: Architecture of J-MOD$^2$. Given an RGB input, features are extracted by the VGG19 module and then fed into the depth estimation and obstacle detection branches to produce dense depth maps and osbtacles bounding boxes.

test on target domains with camera focal lengths that differ from the one used for training.

## IV. Experiments

### A. Datasets

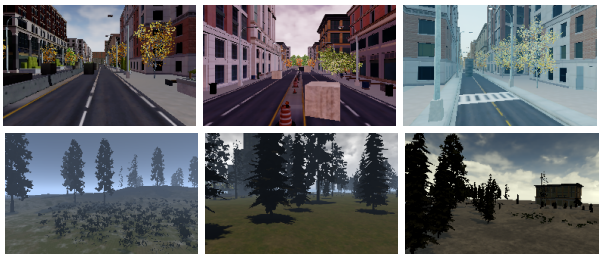


Fig. 3: Sample images of the UnrealDataset.

*1) UnrealDataset:* UnrealDataset is a self-collected synthetic dataset that comprises of more than 100k images and 21 sequences collected in a bunch of highly photorealistic urban and forest scenarios with Unreal Engine and the AirSim plugin [33], which allows us to navigate a simulated MAV inside any Unreal scenarios. The plugin also allows us to collect MAV's frontal camera RGB images, ground truth depth up to 40 meters and segmentation labels. Some samples are shown in Figure 3. We postprocess segmentation labels to form a binary image depicting only two semantic classes: obstacle and non-obstacle by filtering these data with corresponding depth maps, we are finally able to segment obstacles at up to 20 meters from the camera and get ground truth labels for the detection network branch (Fig. 4). MAV's frontal camera has a horizontal field of view of 81,5 degrees.

*2) Zurich Forest Dataset:* Zurich Forest Dataset consist of 9846 real-world grayscale images collected with a stereo camera rig in a forest area. Ground truth depth maps are obtained through semi-global stereo matching [34]. As no segmentation ground truth is available, it is not possible to measure obstacle-wise errors, however we report qualitative results.

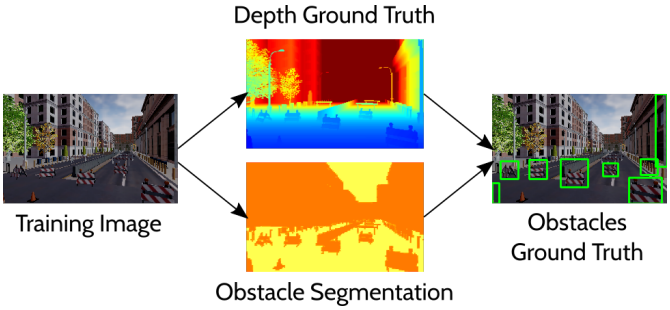


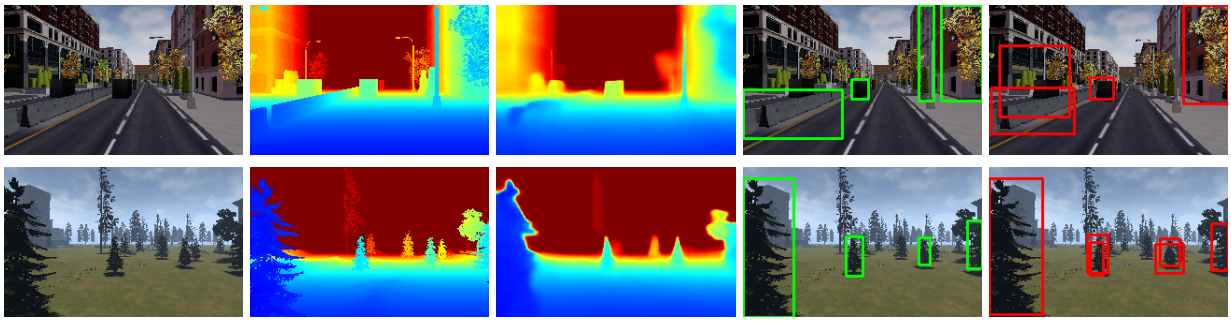

Fig. 4: Given depth and segmentation ground truth, we compute obstacle bounding boxes for each training image. We evaluate only obstacles in a 20 meters range.

### B. Training and testing details

As baselines, we compare J-MOD$^2$ with:

- The depth estimation method proposed in [13].
- Our implementation of the multi-scale Eigen's model [12].
- A simple obstacle detector, consisting of our proposed model, trained without the depth estimation branch.

We train J-MOD$^2$ and all the baseline models on 19 sequences of the UnrealDataset. We left out sequences 09 and 14 for testing. Training is performed with Adam optimizer by setting a learning rate of 0.0001 until convergence. For Eigen's baseline, while we still train on the UnrealDataset, we follow the training procedure described in [12]. Training is performed on a workstation mounting a single NVIDIA Titan X GPU. We release code for J-MOD$^2$ and all the baseline methods at: `http://isar.unipg.it/index.php?option=com_content&view=article&id=47&catid=2&Itemid=188`. We test J-MOD$^2$ on the test sequences of the UnrealDataset and on



(a) RGB Input (b) Depth GT (c) Depth Estimation (d) Obstacle GT (e) Detected Obstacles

Fig. 5: J-MOD$^2$ qualitative results on the UnrealDataset.

| | DEPTH [13] | DETECTOR | EIGEN [12] | J-MOD$^2$ | |
|---|---|---|---|---|---|
| RMSE Full Depth Map | 3.653 | - | 3.785 | **3.473** | Lower |
| Sc.Inv RMSE Full Depth Map | 0.167 | - | 0.166 | **0.155** | is |
| Depth RMSE on Obs.(Mean/Var) | 1.317 / 37.124 | - | 1.854 / 50.71 | **1.034 / 29.583** | better |
| Detection RMSE on Obs.(Mean/Var) | - | 1,534 / 38,568 | - | **1.159 / 30.647** | |
| Detection IOU | - | 56.21% | - | **61.55%** | Higher |
| Detection Precision | - | 61.41% | - | **69.32%** | is |
| Detection Recall | - | 75.83% | - | **79.76%** | better |

TABLE I: Results on the UnrealDataset. For the depth estimation task we report full depth map RMSE and scale invariant errors, obstacle-wise depth and detection branches statistics (mean/variance) estimation errors and detector's IOU, precision and recall.

the whole Zurich Forest Dataset. Note that, while testing on the latter, we do not perform any finetuning.

To evaluate the depth estimator branch performance, we compute the following metrics:

- Linear RMSE and Scale Invariant Log RMSE ($\frac{1}{n}\sum_i d_i^2 - \frac{1}{n^2}(\sum_i d_i)^2$, with $d_i = \log y_i - \log y_i^*$) on the full depth map.
- Depth RMSE on Obstacles (Mean/Variance): For each ground truth obstacle, we compute its depth statistics (mean and variance) and we compare them against the estimated ones by using linear RMSE.

For the detector branch, we compute the following metrics:

- Detection RMSE on Obstacles (Mean/Variance):For each detected obstacle, we compare its estimated obstacle depth statistics (mean and variance) with the closest obstacle ones by using linear RMSE.
- Intersection Over Union (IOU)
- Precision/Recall

### C. Test on UnrealDataset

We report results on the UnrealDataset on Table I. Results confirm how J-MOD$^2$ outperforms the single-task baselines in all metrics, corroborating our starting claim: object structures learned by the detector branch improve obstacles depth estimations of the depth branch. At the same time, localization and accuracy of the detected bounding boxes improve significantly compared to a single-task obstacle detector. Qualitative results are shown on Figure 5.

### D. Testing robustness to focal length

To validate our proposed depth correction strategy introduced in Section III-A, we simulate focal length alterations by cropping and upsampling a central region of the input images of the UnrealDataset. We evaluate performances on different sized crops of images on the sequence-20, comprising of more than 7700 images (see Figure 6). It is worth mentioning that sequence-20 is one of the training sequences. We choose to stage this experiment on a training sequence rather than on a test one to minimize appearance-induced error and make evident the focal-length-induced error. We report results on Table II. When no crop is applied, camera intrinsics are unaltered and appearance-induced error is very low, as expected. As correction is applied linearly on the whole depth map, when scale-dependant error is absent or low, such correction worsen estimations by $19\%$ on non-cropped images. A $230 \times 144$ crop simulates a slightly longer focal length. All metrics worsen, as expected, and correction still cause a $15\%$ higher RMSE error. When $204 \times 128$ crops are evaluated, correction starts to be effective, improving performances by $1,45\%$ with respect to the non-corrected estimation. On $154 \times 96$ crops, correction leads to a $23\%$ improvement. On $128 \times 80$ crops, correction improves performance by $25\%$. We also observe how the detection branch outperforms the depth estimation branch on obstacle distance evaluation as we apply wider crops to the input. This results uphold our hypothesis that detection branch is more robust to large mismatches between training and test camera focal lengths and can be used to partially compensate the induced

Fig. 6: Different-sized crops used to evaluate robustness to focal length changes. From left to right: original sized image, 230x144 crop, 204x128 crop, 154x96 crop, 128x80 crop.

| | ORIGINAL SIZE | | CROP 230X144 | | CROP 204X128 | | CROP 154X96 | | CROP 128X80 | |
|---|---|---|---|---|---|---|---|---|---|---|
| | Cor | NoCor | Cor | NoCor | Cor | NoCor | Cor | NoCor | Cor | NoCor |
| RMSE Full Depth Map | **2.179** | 2.595 | **2.632** | 3.042 | 4.052 | **3.991** | 8.098 | **6.234** | 10.825 | **8.045** |
| Sc. Inv RMSE Full Depth Map | **0.096** | 0.115 | **0.121** | 0.134 | 0.173 | **0.164** | 0.274 | **0.217** | 0.305 | **0.250** |
| Depth RMSE on Obs.(Mean) | **0.185** | 0.676 | **1.293** | 1.458 | 2.465 | **2.219** | 4.865 | **3.583** | 6.148 | **4.485** |
| Detector RMSE on Obs.(Mean) | 0.404 | | 1.079 | | 1.998 | | 4.124 | | 5.450 | |

TABLE II: Results of J-MOD$^2$ on the sequence-20 of the UnrealDataset on different-sized central crops. For each crop, we report in bold the better estimation between unchanged (labeled as NoCor) and corrected depths (labeled as WithCor).

absolute scale estimation deterioration.

### *E. Test: Zurich Forest Dataset*

| | DEPTH [13] | | EIGEN [12] | | J-MOD$^2$ | |
|---|---|---|---|---|---|---|
| | Cor | NoCor | Cor | NoCor | Cor | NoCor |
| RMSE | - | 15.475 | - | 16.410 | **9.125** | 14.333 |
| Sc. Inv RMSE | - | 0.278 | - | **0.273** | 0.277 | 0.277 |

TABLE III: Results on the Zurich Forest Dataset.

Intrinsic parameters of this dataset do not match the UnrealDataset ones, causing large scale-induced errors. Therefore, we can evaluate the performance of J-MOD$^2$ corrected depth, as introduced in Section III-A. In addition, we evaluate the performances of the Eigen's method and the baseline depth estimator. Results are reported on Table III. J-MOD$^2$ outperforms other baselines on the RMSE even without correction, while Eigen still performs slightly better in terms of scale invariant log RMSE. Nevertheless, correction factor improves J-MOD$^2$ depth's RMSE by $36\%$.

### *F. Navigation experiments*

We further validate J-MOD$^2$ effectivness for obstacle detection applications by setting up a simulated full MAV navigation system. We depict the system architecture in Figure 8. We create a virtual forest scenario on Unreal Engine, slightly different from the one used for dataset collection. The line-of-sight distance between the takeoff point and the designed landing goal is about $61$ meters. Trees are about $6$ meters tall and spaced $7$ meters from each other, on average. An aerial picture of the test scenario is reported in Figure 9.

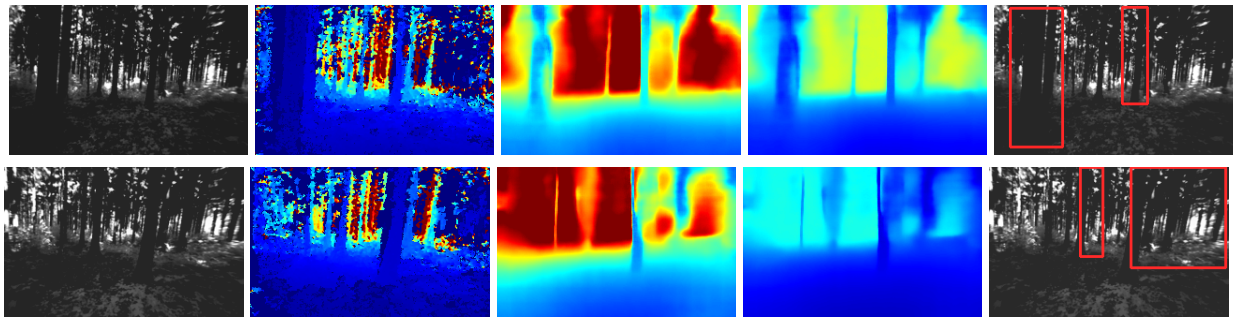
A simulated MAV is able to navigate into the scenario and collect RGB images from its frontal camera. We estimate depth from the captured input and we employ it to dynamically build and update an Octomap [35]. We plan obstacle-free trajectories exploiting an off-the shelf implementation of the RRT-Connect planner [36] from the *MoveIt!* ROS library, which we use to pilot the simulated MAV at a cruise speed of $1m/s$. Trajectories are bounded to a maximum altitude of $5$ meters. As a new obstacle is detected along the planned trajectory, the MAV stops and a new trajectory is computed. The goal point is set $4$ meters above the ground. For each flight, we verify its success and measure the flight distance and duration. A flight fails if the MAV crashes or gets stuck, namely not completing its mission in a $5$ minute interval. We compare J-MOD$^2$ with the Eigen's baseline, both trained on the UnrealDataset.

While planning, we add a safety padding on each Octomap obstacles. This enforces the planner to compute trajectories not too close to the detected obstacles. For each estimator, we set this value equal the average RMSE obstacle depth error on the UnrealDataset test set, as reported in Table I: $1.034$ meters for J-MOD$^2$, $1.854$ meters for Eigen. We refer to this value as a reliability measure of each estimator; the less accurate an estimator is, the more padding we need to guarantee safe operation. We perform 15 flights for each depth estimator and report their results on Table IV.

| | EIGEN [12] | J-MOD$^2$ |
|---|---|---|
| Success rate | 26,6% | **73,3%** |
| Failure cases | 8 stuck / 3 crash | 2 stuck / 2 crash |
| Avg. flight time | 147s | **131s** |
| Std. Dev. Flight Time | 18.51s | **12.88s** |
| Avg. flight distance | 78m | **77m** |
| Std. Dev. Flight Distance | **4.47m** | 9.95m |

TABLE IV: Results of the navigation experiment. We compare the navigation success rate when using J-MOD$^2$ and Eigen's approach as obstacle detection systems

J-MOD$^2$ clearly performs better in all metrics, proving that how our method is effective for monocular obstacle detection. By analyzing failure cases, for 6 times the MAV



(a) RGB Input (b) Depth GT (c) Non Corrected Depth Estimation (d) Corrected Depth Estimation (e) Detected Obstacles

Fig. 7: J-MOD$^2$ qualitative results on the Zurich Forest Dataset

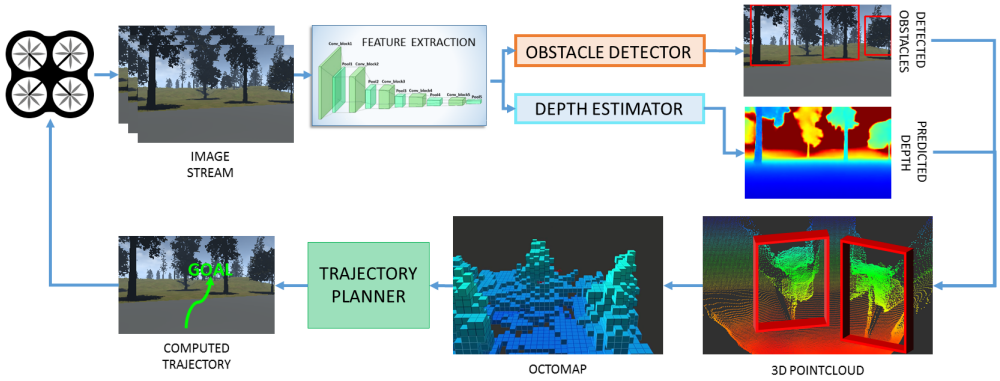



Fig. 8: Architecture of the full navigation pipeline. For each RGB image captured by the MAV frontal camera, a depth map is computed and converted into a point cloud used to update the 3D map and compute an obstacle-free trajectory. The MAV then flies along the computed trajectory until a new obstacle is detected.

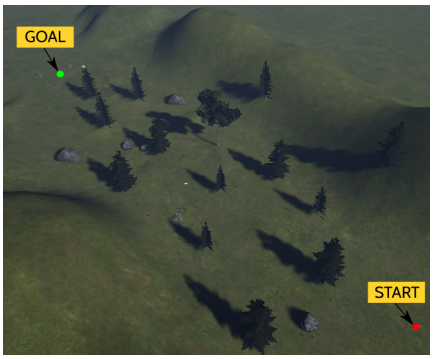



Fig. 9: Navigation test scenario. Line-of-sight distance between start(bottom right) and goal(top left) points is about 61 meters.

using Eigen as obstacle detector got stuck in the proximity of goal point because ground was estimated closer than its real distance, causing planner failure in finding an obstacle-free trajectory to the goal. J-MOD$^2$ failures are mostly related on erratic trajectory computation which caused the MAV to fly too close to obstacles, causing lateral collisions or getting stuck in proximity of tree's leaves.

## V. Conclusion and Future Work

In this work, we proposed J-MOD$^2$, a novel end-to-end deep architecture for joint obstacle detection and depth estimation. We demonstrated its effectiveness in detecting obstacles on synthetic and real-world datasets. We tested its robustness to appearance and camera focal length changes. Furthermore, we deployed J-MOD$^2$ as an obstacle detector and 3D mapping module in a full MAV navigation system and we tested it on a highly photo-realistic simulated forest scenario. We showed how J-MOD$^2$ dramatically improves mapping quality in a previously unknown scenario, leading to a substantial lower navigation failure rate than other SotA depth estimators. In future works, we plan to further improve robustness over appearance changes, as this is the major challenge for the effective deployment of these algorithms in practical real-world scenarios.

## References

[1] A. Bachrach, S. Prentice, R. He, and N. Roy, "Range–robust autonomous navigation in gps-denied environments," *Journal of Field Robotics*, vol. 28, no. 5, pp. 644–666, 2011.

[2] S. Grzonka, G. Grisetti, and W. Burgard, "A fully autonomous indoor quadrotor," *IEEE Transactions on Robotics*, vol. 28, no. 1, pp. 90–100, 2012.

[3] F. Fraundorfer, L. Heng, D. Honegger, G. H. Lee, L. Meier, P. Tanskanen, and M. Pollefeys, "Vision-based autonomous mapping and exploration using a quadrotor mav," in *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*. IEEE, 2012, pp. 4557–4564.

[4] C. De Wagter, S. Tijmons, B. D. Remes, and G. C. de Croon, "Autonomous flight of a 20-gram flapping wing mav with a 4-gram onboard stereo vision system," in *Robotics and Automation (ICRA), 2014 IEEE International Conference on*. IEEE, 2014, pp. 4982–4987.

[5] A. Bachrach, S. Prentice, R. He, P. Henry, A. S. Huang, M. Krainin, D. Maturana, D. Fox, and N. Roy, "Estimation, planning, and mapping for autonomous flight using an rgb-d camera in gps-denied environments," *The International Journal of Robotics Research*, vol. 31, no. 11, pp. 1320–1343, 2012.

[6] A. S. Huang, A. Bachrach, P. Henry, M. Krainin, D. Maturana, D. Fox, and N. Roy, "Visual odometry and mapping for autonomous flight using an rgb-d camera," in *Robotics Research*. Springer, 2017, pp. 235–252.

[7] R. A. Newcombe, S. J. Lovegrove, and A. J. Davison, "Dtam: Dense tracking and mapping in real-time," in *Computer Vision (ICCV), 2011 IEEE International Conference on*. IEEE, 2011, pp. 2320–2327.

[8] M. W. Achtelik, S. Lynen, S. Weiss, M. Chli, and R. Siegwart, "Motion-and uncertainty-aware path planning for micro aerial vehicles," *Journal of Field Robotics*, vol. 31, no. 4, pp. 676–698, 2014.

[9] D. Scaramuzza, M. C. Achtelik, L. Doitsidis, F. Friedrich, E. Kosmatopoulos, A. Martinelli, M. W. Achtelik, M. Chli, S. Chatzichristofis, L. Kneip, *et al.*, "Vision-controlled micro flying robots: from system design to autonomous navigation and mapping in gps-denied environments," *IEEE Robotics & Automation Magazine*, vol. 21, no. 3, pp. 26–40, 2014.

[10] J. Engel, T. Schöps, and D. Cremers, "Lsd-slam: Large-scale direct monocular slam," in *European Conference on Computer Vision*. Springer, 2014, pp. 834–849.

[11] C. Forster, M. Pizzoli, and D. Scaramuzza, "Svo: Fast semi-direct monocular visual odometry," in *Robotics and Automation (ICRA), 2014 IEEE International Conference on*. IEEE, 2014, pp. 15–22.

[12] D. Eigen and R. Fergus, "Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 2650–2658.

[13] M. Mancini, G. Costante, P. Valigi, T. A. Ciarfuglia, J. Delmerico, and D. Scaramuzza, "Towards domain independence for learning-based monocular depth estimation," *IEEE Robotics and Automation Letters*, 2017.

[14] S. Yang, S. Konam, C. Ma, S. Rosenthal, M. Veloso, and S. Scherer, "Obstacle avoidance through deep networks based intermediate perception," *arXiv preprint arXiv:1704.08759*, 2017.

[15] L. Heng, D. Honegger, G. H. Lee, L. Meier, P. Tanskanen, F. Fraundorfer, and M. Pollefeys, "Autonomous visual mapping and exploration with a micro aerial vehicle," *Journal of Field Robotics*, vol. 31, no. 4, pp. 654–675, 2014.

[16] C. Nous, R. Meertens, C. De Wagter, and G. de Croon, "Performance evaluation in obstacle avoidance," in *Intelligent Robots and Systems (IROS), 2016 IEEE/RSJ International Conference on*. IEEE, 2016, pp. 3614–3619.

[17] A. J. Barry and R. Tedrake, "Pushbroom stereo for high-speed navigation in cluttered environments," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, May 2015, pp. 3046–3052.

[18] A. J. Barry, H. Oleynikova, D. Honegger, M. Pollefeys, and R. Tedrake, "Fpga vs. pushbroom stereo vision for mavs," in *Vision-Based Control and Navigation of Small Lightweight UAVs, IROS Workshop*, 2015.

[19] M. Pizzoli, C. Forster, and D. Scaramuzza, "Remode: Probabilistic, monocular dense reconstruction in real time," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2014, pp. 2609–2616.

[20] H. Alvarez, L. M. Paz, J. Sturm, and D. Cremers, "Collision avoidance for quadrotors with a monocular camera," in *Experimental Robotics*. Springer, 2016, pp. 195–209.

[21] A. Beyeler, J.-C. Zufferey, and D. Floreano, "Vision-based control of near-obstacle flight," *Autonomous robots*, vol. 27, no. 3, pp. 201–219, 2009.

[22] C. Bills, J. Chen, and A. Saxena, "Autonomous mav flight in indoor environments using single image perspective cues," in *Robotics and automation (ICRA), 2011 IEEE international conference on*. IEEE, 2011, pp. 5776–5783.

[23] T. Mori and S. Scherer, "First results in detecting and avoiding frontal obstacles from a monocular camera for micro unmanned aerial vehicles," in *Robotics and Automation (ICRA), 2013 IEEE International Conference on*. IEEE, 2013, pp. 1750–1757.

[24] A. Giusti, J. Guzzi, D. C. Cireşan, F.-L. He, J. P. Rodríguez, F. Fontana, M. Faessler, C. Forster, J. Schmidhuber, G. Di Caro, *et al.*, "A machine learning approach to visual perception of forest trails for mobile robots," *IEEE Robotics and Automation Letters*, vol. 1, no. 2, pp. 661–667, 2016.

[25] D. Eigen, C. Puhrsch, and R. Fergus, "Depth map prediction from a single image using a multi-scale deep network," in *Advances in neural information processing systems*, 2014, pp. 2366–2374.

[26] F. Liu, C. Shen, G. Lin, and I. Reid, "Learning depth from single monocular images using deep convolutional neural fields," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 38, no. 10, pp. 2024–2039, Oct 2016.

[27] P. Chakravarty, K. Kelchtermans, T. Roussel, S. Wellens, T. Tuytelaars, and L. Van Eycken, "Cnn-based single image obstacle avoidance on a quadrotor," in *Robotics and Automation (ICRA), 2017 IEEE International Conference on*. IEEE, 2017, pp. 6369–6374.

[28] P. K. Nathan Silberman, Derek Hoiem and R. Fergus, "Indoor segmentation and support inference from rgbd images," in *ECCV*, 2012.

[29] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The kitti dataset," *The International Journal of Robotics Research*, p. 0278364913491297, 2013.

[30] C. Cadena, Y. Latif, and I. D. Reid, "Measuring the performance of single image depth estimation methods," in *Intelligent Robots and Systems (IROS), 2016 IEEE/RSJ International Conference on*. IEEE, 2016, pp. 4150–4157.

[31] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 779–788.

[32] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[33] S. Shah, D. Dey, C. Lovett, and A. Kapoor, "Airsim: High-fidelity visual and physical simulation for autonomous vehicles," *arXiv preprint arXiv:1705.05065*, 2017.

[34] H. Hirschmuller, "Stereo processing by semiglobal matching and mutual information," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 30, no. 2, pp. 328–341, 2008.

[35] A. Hornung, K. M. Wurm, M. Bennewitz, C. Stachniss, and W. Burgard, "Octomap: An efficient probabilistic 3d mapping framework based on octrees," *Autonomous Robots*, vol. 34, no. 3, pp. 189–206, 2013.

[36] J. J. Kuffner and S. M. LaValle, "Rrt-connect: An efficient approach to single-query path planning," in *Robotics and Automation, 2000. Proceedings. ICRA'00. IEEE International Conference on*, vol. 2. IEEE, 2000, pp. 995–1001.