



# Robust Path Planning in the Wild for Automatic Look-Ahead Camera Control

Sander R. Klomp<sup>1,2</sup> <sup>a</sup> and Peter H. N. de With<sup>1</sup> <sup>b</sup>

<sup>1</sup>*Department of Electrical Engineering, Eindhoven University of Technology, Eindhoven, The Netherlands*

<sup>2</sup>*ViNotion B.V., Eindhoven, The Netherlands*

**Keywords:** Semantic Segmentation, Deep Learning, Road Segmentation, Path Planning.

**Abstract:** Finding potential driving paths on unstructured roads is a challenging problem for autonomous driving and robotics applications. Although the rise of autonomous driving has resulted in massive public datasets, most of these datasets focus on urban environments and feature almost exclusively paved roads. To circumvent the problem of limited public datasets of unpaved roads, we combine seven public vehicle-mounted-camera datasets with a very small private dataset and train a neural network to achieve accurate road segmentation on almost any type of road. This trained network vastly outperforms networks trained on individual datasets when validated on our unpaved road datasets, with only a minor performance reduction on the highly challenging public WildDash dataset, which is mostly urban. Finally, we develop an algorithm to robustly transform these road segmentations to road centerlines, used to automatically control a vehicle-mounted PTZ camera.

## 1 INTRODUCTION

Road segmentation and path planning are two essential components of autonomous driving systems. In recent years, large performance gains over conventional computer vision-based systems have been realized using deep learning, allowing for self-driving vehicles on most public roads. However, several corner cases remain in which these well-established methods cause failures (Grigorescu et al., 2020), especially when the perception part of the algorithm cannot segment the drivable surface correctly.


One of the cases where modern neural network-based solutions struggle with path planning is on unstructured roads, where a lack of lane markings can confuse lane detection networks or networks that directly perform the path-planning task. This problem is exacerbated by the fact that the large public datasets for autonomous driving focus on urban environments where most roads are structured, such as BDD100k (Yu et al., 2020) and Cityscapes (Cordts et al., 2016). The Mapillary Vistas (Neuhold et al., 2017) and IDD datasets (Varma et al., 2019) include some types of unstructured roads, but limit their variation to paved roads. For robust path planning in any environment, we argue that unpaved roads, such as


sand, dirt or even just tire tracks, should be included also.

This research focuses on developing a robust path-planning system for any type of road. The road segmentation is performed using a HRNetV2-W48 network (Sun et al., 2019) trained on a combination of eight different datasets, in order to maximize the variation of learned road types. Additional post-processing is introduced on the road segmentations to further improve the robustness. Because existing datasets are not annotated with paths, we fall back to a conventional computer vision approach to perform the final path planning, which is consequently based on the segmentation masks. Whereas most of the algorithms in this paper can be also applied to autonomous driving, the more specific use case for which this work has been performed, is different. Our main use case is a person-driven vehicle with a mounted Pan-Tilt-Zoom (PTZ) camera, where the camera view should automatically stay concentrated at the center of the road at a certain distance in front of the vehicle, thereby actively following any curves in the road. Several design decisions in the path-planning algorithm are specific to this use case, though most of the algorithms may also be applied for collision avoidance. Some example outputs of the algorithm are shown in Fig. 1.

Our contributions are summarized as follows.

1. We combine several urban and non-urban datasets

<sup>a</sup>  <https://orcid.org/0000-0002-0874-4720>

<sup>b</sup>  <https://orcid.org/0000-0002-7639-7716>

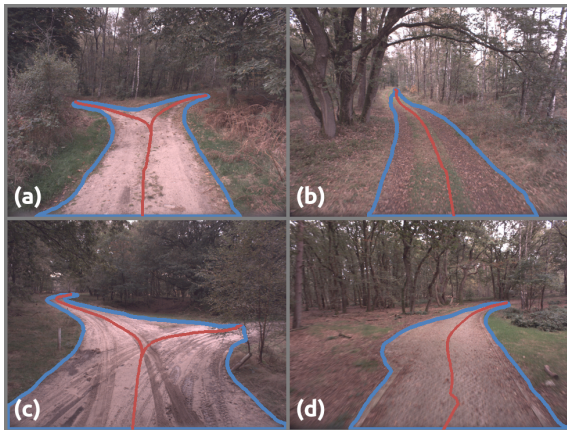


Figure 1: Pathfinding examples. Road segmentations are outlined in blue, estimated centerlines in red.

into a diverse, new large dataset, to maximize the robustness of a trained segmentation network, allowing it to segment nearly any type of road.

2. A novel path-planning algorithm is developed that transforms the road segmentation into road centerlines, to be used to control a forward-looking vehicle-mounted PTZ camera.

## 2 RELATED WORK

The path-planning system has related work in both neural networks for semantic segmentation and general path planning, which are discussed separately.

### 2.1 Semantic Segmentation

Early semantic segmentation research using neural networks, such as the FCN (Shelhamer et al., 2016), focused on general object segmentation. Since then, two main improved architectures have become popular: the encoder-decoder structure with skip connections, popularized by the U-net (Ronneberger et al., 2015), and the structures that maintain high-resolution features throughout the entire network, such as HRNet (Sun et al., 2019). Both types of networks reach state-of-the-art performance on various semantic segmentation datasets.

With the emerging application of autonomous driving, more public driving segmentation datasets have become available. These datasets include Cityscapes (Cordts et al., 2016), IDD (Varma et al., 2019), BDD100K (Yu et al., 2020) and Mapillary Vistas (Neuhold et al., 2017), which all focus on segmentation in urban environments.

Many works focus on achieving the highest pos-

sible accuracy on these public urban datasets, but we are interested specifically in generalizing performance on non-urban settings with unstructured roads. Several works do focus on the detection of unstructured roads, such as (Nakagomi et al., 2021), (Yadav et al., 2017), (Giusti et al., 2016), (Valada et al., 2016), but they either require LIDAR (Nakagomi et al., 2021), use heuristics that are sensitive to color differences on the road such as shadows (Yadav et al., 2017), or train on a small dataset with very little road variation ((Giusti et al., 2016) and (Valada et al., 2016)). Several datasets also exist containing natural or rural environments, such as YCOR (Maturana et al., 2018), RUGD (Wigness et al., 2019), Freiburg Forest (Valada et al., 2016) and Rellis-3D (Jiang et al., 2020), but all of these datasets show little variety in road types and environments. To our knowledge, neither a comprehensive dataset nor a robust road segmentation method exists yet that yields an acceptable performance on general unstructured road. Because of this, our road segmentation will be performed by training on a combination of the aforementioned unpaved road datasets.

### 2.2 Path Planning

For regular autonomous driving, path planning generally consists of four steps: route planning, behavioral decision-making, motion planning and vehicle control (Paden et al., 2016). Considering that our use case is only to control the camera viewing direction instead of automated driving of the entire vehicle, most of these steps can be simplified or even omitted. “Route planning” is somewhat relevant, because at intersections, the system should choose the correct road to follow. “Behavioral decision-making” is unnecessary, because the human driver determines the route and the camera does not need to react in any way to other road users. “Motion planning” involves determining the detailed path based on the current view of the road, and is the primary focus of this work. “Vehicle control”, which becomes camera control in our case, is trivial: rotating a simple PTZ camera is much simpler than steering a vehicle and does not require complex vehicle models.

Motion planning can be performed in several ways. The most straightforward methods are graph search-based planners, such as the Dijkstra and A\* algorithms, although planners based on sampling such as Rapidly-exploring Random Trees (RRT) are also common (González et al., 2016). More complex methods aim to either be faster to compute than A\*, allow for easier integration of other parameters such as vehicle size, or provide smoother paths which en-

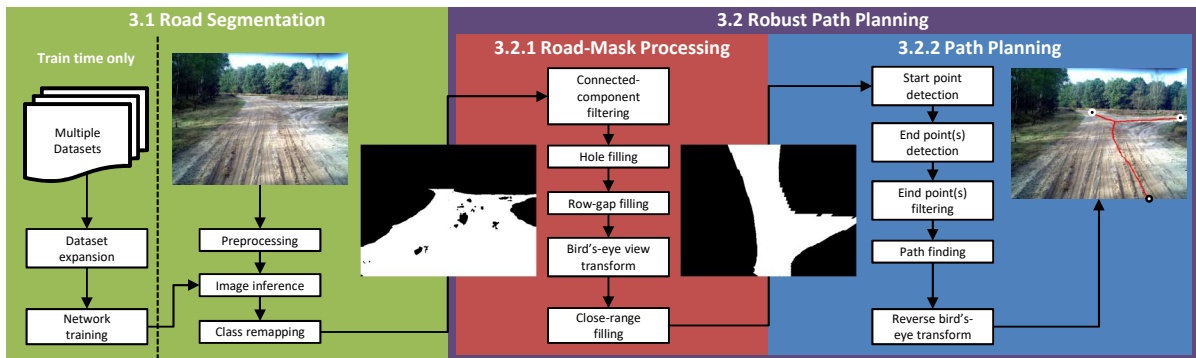


Figure 2: Flow chart of the path-planning system. Note that a poorly trained network is used deliberately for the segmentation to highlight the effect of road-mask processing. Road path, start point and end point are shown in the right image.

able smoother vehicle control. For example, motion planning can be performed far more efficiently using a neural network (Qureshi et al., 2019), by teaching it to imitate paths generated using RRT. This is especially useful in high-dimensional spaces and not necessary for our simple 2D path-planning case. To conclude, we will perform path planning using the simple A\* algorithm, because it is sufficiently fast for our use case and the additional features of more complex methods are not necessary for camera control.

### 3 CREATING A ROBUST PATH-PLANNING SYSTEM

The proposed method consists of three main components, as shown in Fig. 2: semantic segmentation, road-mask processing and path planning. For semantic segmentation, the focus is on how to combine existing datasets and jointly train on them to achieve strong generalization. During inference, the semantic segmentation stage simply consists of executing the network and mapping several classes to a road mask. The road mask is then post-processed using multiple conventional computer vision algorithms, to ensure better pathfinding in the final stage. Pathfinding itself employs the A\* algorithm, although automatically finding the correct start and end points for that algorithm is not always straightforward. These components are now further discussed in detail.

#### 3.1 Road Segmentation: Multi-Dataset Training for Improved Robustness

This section first briefly describes the employed CNN and the applied training procedure, followed by an overview of suitable datasets and the proposed method for combining these datasets.

The base network that is employed for road segmentation is the HRNetV2-W48 network (Sun et al., 2019). This neural network does not follow the common encoder-decoder structure, but instead maintains a full-resolution branch throughout the entire network and several downsampled branches in parallel for enlarging the receptive field. To finally combine the separate branches, the downsampled branches are upsampled and concatenated to the main branch (Sun et al., 2019). Whereas optimizing this architecture further to improve the robustness of our road segmentation task may be possible, we instead adopt the idea from MSeg (Grigorescu et al., 2020) and focus on the data instead of the CNN architecture.

The authors from MSeg have already shown that an unaltered HRNetV2 trained on seven different datasets simultaneously outperforms robustness-specialized techniques on the highly challenging and diverse WildDash (Zendel et al., 2018) test set. However, Mseg does not yet include datasets with unpaved roads. Furthermore, while WildDash is highly varied from a weather and environments point of view, it still primarily focuses on urban scenes, making it less suited as a test set for our work. Because of this, we will only employ WildDash to verify generalization of our trained network to urban scenes.

To perform dataset balancing, we adopt the method from Mseg: in every training batch an equal number of images from each dataset is used. For example, this means that if there is one dataset of 10,000 images and one of 1,000 images, the former will only be seen by the network once per epoch, while the latter will be seen ten times per epoch. The advantage of this approach is that it forces the network to perform well on all datasets, but the downside is that it may also cause overfitting on small datasets or datasets with low variety. We expect overfitting on the small datasets to not be a problem, due to the large number of different datasets that we

will be combining, but this aspect should certainly be kept in mind when combining fewer datasets.

**Dataset Expansion.** The training datasets that MSeg employs are extremely varied: two datasets containing all kinds of scenes, both indoor and outdoor, with various objects (COCO and ADE20K), the indoor segmentation dataset SUN RGBD, and the four driving datasets Mapillary Vistas, IDD, BDD100K and Cityscapes. MSeg also tests on highly varied datasets, but we limit ourselves to the testing of road segmentation, to achieve stronger performance on our specialized task: robust road segmentation on unstructured and unpaved roads.

#### A. Public Off-Road Dataset Additions.

Several small segmentation datasets exist that may contain useful data for unpaved road segmentation. All datasets are filtered to only contain images with at least one pixel labeled as “road”. The number of images left after filtering is given between brackets.

- Yamaha-CMU Off-Road (YCOR) dataset (Maturana et al., 2018): 1,076 (982) labeled images collected at four different locations using a vehicle-mounted camera, with quite variable road types: concrete, dirt, gravel and tire tracks, providing a good baseline for unpaved road segmentation.
- Robot Unstructured Driving Ground (RUGD) dataset (Wigness et al., 2019): 7,453 (3,637) labeled images in eight different terrains, recorded using a small robot. The dataset contains mostly asphalt and some gravel roads. However, the roads are clearly not the focus of the data collection, as the robot mostly drives besides the roads and paths. The size of the robot also causes the camera to be very close to the ground.
- Freiburg Forest (FF) dataset (Valada et al., 2016): 366 (366) labeled images in forest environments, recorded using both an infrared camera and a regular RGB camera. This data are highly relevant to our use case, even though the set is relatively small and contains limited variety. Most roads in the dataset are gravel-like.
- Rellis-3D dataset (Jiang et al., 2020): A dataset with 6,235 (3,906) images, recorded in a grass field with dirt tire tracks. Although the dataset is quite large, the image variety is extremely limited.

We remap all the labels of these four datasets and combine them with the four public urban driving datasets, for a total of eight different training datasets.

#### B. Semi-Supervised Addition of New Data

To increase the variety of data further, additional data are recorded and annotated. Data are recorded

on several road types: sand, dirt, tire tracks, asphalt, concrete and stone-paved roads, mostly in forest environments. Since the focus is on road segmentation and the creation of full-image semantic labels is extremely time-consuming, we only manually label the road class. All non-road pixels in the dataset are (pseudo-)labeled automatically using the HRNetV2-W48, which is trained on a combination of all Mseg datasets that contain roads (Mapillary Vistas, BDD100k, Cityscapes and IDD) and the four public unpaved road datasets (YCOR, RUGD, FF and Rellis3D). Since inference time is irrelevant for one-time labeling, we improve the pseudo-labels by applying the common trick of performing inference on both the original and mirrored version of the same image at multiple scales and averaging the predictions to obtain the final pseudo-labeled image.

This procedure results in 114 annotated images that were recorded in one location in forests in The Netherlands, another 283 images spread over multiple locations in The Netherlands and 116 images that were captured in forests and fields in Austria. The recordings were performed over several days, so that there is some weather variety, though it is mostly overcast and sunny. The 114 images of the first location in The Netherlands are added to the training data (NL), the 283 images of other locations in the Netherlands and 116 Austria images are used as validation sets NL-val and AT-val, respectively.

## 3.2 Robust Path Planning

The primary goal of the system is to find the center of the road, or all possible road centers in all directions in the case of crossings. In general, the path-planning algorithm consists of the following steps: the semantic output of the neural network is converted to a binary road mask, then transformed into a bird’s-eye view (BEV), and finally used for the path planning, by detecting start and end points of the roads and drawing paths between these points.

Although this general method is adequate to detect the road centerline in ideal conditions, there are several common cases that result in less ideal paths without additional filtering. These cases are:

- Poor road segmentation. In most cases, this occurs because parts of the road are mapped to road-like classes, such as “gravel” instead of the specific “road” class, or sometimes road pixels are mapped to entirely different classes if the domain of the test image is too different from the CNN training data. Both cases can cause holes in the road mask.

- Perspective transform distortion of far-away pixels. After perspective transformation, far-away pixels become very large in the BEV road mask, which can cause separate road detections on tiny bumps in the original road mask.
- Poor close-range road centerline accuracy. Due to a limited horizontal camera field of view, the entire width of the road will not be captured near the camera position. If the vehicle is not in the center of the road, this causes an incorrect path-planning start point.
- Detected end points not at the road center. For example, for a perfectly straight road, BFS will find the largest distance to the start point at the top-left and top-right corners of the road, instead of the road center.

The following sections describe the implementation and improvement of the proposed path-planning algorithms to significantly reduce the aforementioned problems, structured according to the Road Mask Processing and Path Planning blocks of Fig. 2.

### 3.2.1 Road-Mask Processing

Compared to simply using the binary road mask resulting from segmentation directly, several improvements are proposed for more robust path generation.

*A. Class Remapping.* To convert the semantic segmentation map into a road mask, it is possible to remap multiple classes of the output of the neural network to road, instead of only the “road” class itself. The classes that generally still match road are: “gravel”, “railroad” (especially in the presence of tram rails), and “runway”. In practice, we have noticed that it can help to also remap “snow” to road, as some sandy roads are misclassified into snow, although clearly this is only viable if the system will never be used in an environment containing actual snow. Note that this remapping is performed at inference time only and not during training of the segmentation CNN. Remapping during training would require manual relabeling, especially for the “gravel” class, because this class sometimes refers to terrain and sometimes to gravel-like roads in the public datasets.

*B. Connected-Component Filtering.* We assume that in normal operating conditions, there can only be a single “road” blob. Hence, small spurious road detections are removed by finding the largest connected component and setting all other mask pixels to black.

*C. Hole Filling.* A normal road will not have areas of non-road inside it, hence holes are filled by finding the contours of all blobs of the road mask and filling the inner contours. Note that this step is only viable for

camera control, where the center of the road should be found regardless of objects on the road. In the case of autonomous driving, this step should be omitted, because holes in the road mask likely indicate objects that should be avoided. In Fig. 3 a single hole is filled, shown in green.

*D. Row-Gap Filling.* When using a perspective transform to compute a BEV, far-away pixels will cover a larger area of the BEV road mask than nearby ones. This causes strong jagged edges in the upper part of the road mask, which is harmful for road end-point detection. A simple heuristic for circumventing this problem, is filling up small (up to a few pixels) horizontal gaps at the edge of the road mask, prior to computing the BEV. An example of this issue and why filling horizontal row gaps resolves the jagged edges, is shown in Fig. 3 and highlighted in red.

*E. Bird’s-Eye View Transform.* To simplify the path-planning process, the road mask is converted to BEV using a perspective transformation. The perspective transformation parameters are estimated based on a single image from the vehicle-mounted camera setup on a long flat straight road, where a vanishing point is clearly visible. Note that this calibration is only valid for a fixed tilt and zoom setting of the PTZ camera and will result in an increasingly incorrectly warped result in case the road surface is not flat. Thankfully, in practice, the path-planning algorithms are quite robust against incorrect perspective transformations because of the inversion of the transform at the end of the path-planning stage.

*F. Close-Range Filling.* Due to the limited camera field of view, the road area that is very close to the vehicle is normally out of view. When the vehicle is not currently positioned at the center of the road, the road centerline estimation becomes inaccurate when close to the vehicle. Filling the BEV downwards from the point where the detected road touches the edges of the original image, solves this problem. An example is shown in Fig. 3, at the bottom of the BEV road mask, where the filled area is indicated in blue.

### 3.2.2 Path Planning

Path planning consists of finding the shortest possible paths from a start point to one or several end points, as visualized in the rightmost image in the flowchart of Fig. 2. More specifically, the possible paths are constrained to be as close to the road center as possible. Besides simply finding these start and end points, we perform several steps to improve their quality.

*Detection of the Start Point.* To find the path-planning start point, it is assumed that the vehicle is driving on the road. Under this assumption, the middle of the bottom row of “road” pixels in the BEV is a reason-

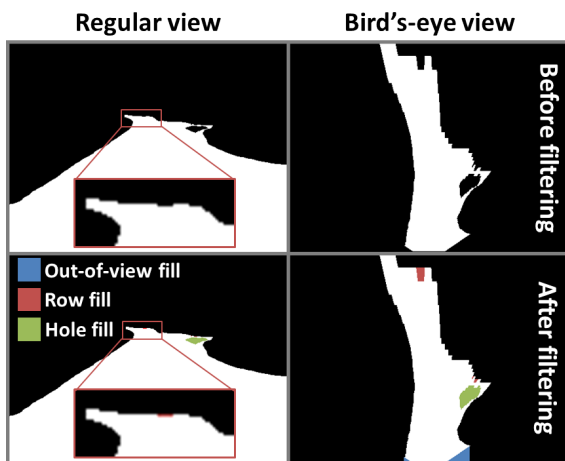


Figure 3: Mask filtering impact example. The impact of row-gap filling is enhanced and highlighted in red at the top of the road, out-of-view filling is shown in blue and hole filling in green.

able starting point from which the path planning is commencing. Note that this point can be slightly in front of the vehicle depending on camera tilt, because the camera is at a certain height above the ground and has limited vertical field of view. Hence, the bottom row of pixels in the original image, which corresponds to the bottom row of pixels in the BEV image, is aligned with the line where the vertical field of view of the camera intersects with the ground plane. In our test setup, the tilt of the PTZ camera is set such that this point is approximately four meters in front of the vehicle.

*Detection of End Point(s).* In order to find all viable road centerlines, end points should be located on all branching roads in the image. For example, an intersection generally results in three viable road centerlines. Potential endpoints are detected by applying a Breadth First Search (BFS) on the BEV road mask. This search returns an image where every pixel has a value representing the distance to the start point. End points are then estimated by finding local maxima in the resulting image. To prevent a large number of spurious maxima at rough edges of the mask, we consider a local maximum only if it is the maximum value within a certain window size, instead of merely comparing to adjacent pixels.

Improving the BFS is one way to achieve more reliable endpoints. We modify the 8-way BFS so that moving diagonally costs  $\sqrt{2}$  movement instead of unity. This improves the detection of end points for side roads or roads with strong curves, as the approximated distance is closer to a Euclidean distance. *End point(s) Filtering.* Using the above algorithm, results in several incorrect end points. The majority

of these incorrect end points are caused by two aspects. First, due to the way that endpoints are detected, namely using BFS, the end points may not end up in the middle of the road, but instead at the corners of the road mask, because they are slightly further away. This only causes the last part of the path to be incorrect, but this issue is still worth addressing. To alleviate this problem, we search for end points at the top row of road mask pixels and merge (in case of two end points connected by only white pixels) or move them (in case of a single end point) to the center of this row of pixels. An example is shown at the top of Fig. 4. The issue is now resolved for straight roads, but end points for curved roads may also be moved incorrectly. To prevent moving end points of curved roads, end points are only moved if their horizontal positions are within the width of the road at the bottom of the BEV, plus a small margin. The second cause of incorrect end points is shape deviations in the road mask, which primarily appear when the road is not clearly separable from the roadside. This happens most often close to the vehicle, where the high resolution allows for example small clumps of grass to be detected as “not road”, as in the bottom right of Fig. 4. In practice, because this issue only appears close to the vehicle, we simply filter out road end points that appear too close to the vehicle. In case there really is a road fork this close to the vehicle, the camera should have been tracking it earlier, and would have rotated its orientation such that the fork would no longer be at the side of the view. Hence, we can conclude that this road fork is not the desired path anyway.



Figure 4: Example of end point filtering which shows both centering end points at road endings and filtering out nearby end points.

*Path Finding.* To find a path through the center of the road from the start point to all end points, a distance transform is applied to the BEV road mask, inverted and then used as the cost for an A\* path-finding algorithm. The low cost far away from non-road pixels causes the A\* algorithm to find a path as far away as possible from these non-road pixels, which approximates the center of the road.

When taking the regular distance transform of an image, at every pixel, the distance to the nearest zero pixel is returned. Local high values in the distance

transform are a good approximation of the center of the road, as long as the road touches both the top and bottom of the BEV image. In cases where distant road is occluded, for example due to a height difference, the road mask in the BEV will end partway in the image. An example of this is shown in Fig. 5. As can be observed at the bottom left of the Figure, this causes the high values in the distance transform to no longer align with the center of the road. To remedy this, the BEV mask is temporarily modified by changing the pixel values of the rows above the top-most road-mask pixel to white, after which the distance transform is applied. The rows of changed pixels are then set back to black in the distance transform, resulting in the bottom-right image of Fig. 5. A similar process can be applied for cases where the road pixels do not start at the bottom-most row of the BEV, although this case is uncommon.

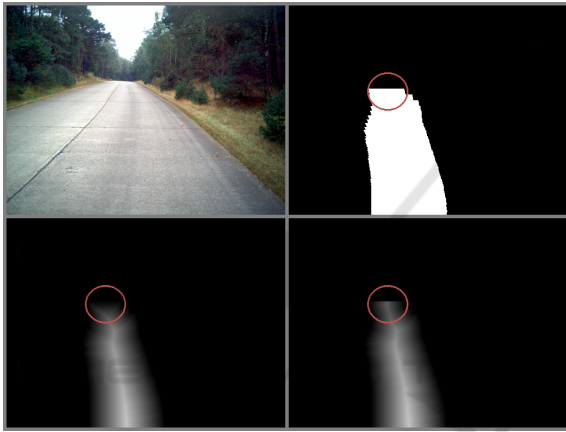


Figure 5: Modified distance transform example. The local maxima of the distance transform now more clearly follow the center of the road, all the way until the end of the road mask.

## 4 EXPERIMENTAL RESULTS

The resulting path-finding system performs well on our private datasets. In the following sections, the road segmentation and path-finding results are discussed separately. For road segmentation, we primarily investigate the impact of training with different datasets.

### 4.1 Impact of Datasets on Performance

The selection of datasets that are used to train the road segmentation model has a large impact on the performance. To evaluate the impact of each dataset on unstructured road segmentation performance, we fine-tune the Mseg-pretrained HRNet-W48 for 10 epochs

on each individual dataset and test the road segmentation performance on our two validation datasets in the Netherlands (NL-val) and Austria (AT-val). The road IoU of the WildDash validation set (WD-val) is reported to verify that the model is not over-trained on only unstructured or unpaved roads, but the performance on this dataset is not leading for our application. The results of fine-tuning on individual datasets are listed in the middle block of Table 1. “Mseg HRNet-W48” refers to the model pre-trained by (Grigorescu et al., 2020) on ADE20K, BDD100K, Cityscapes, COCO-Panoptic, IDD, Mapillary Vistas and SUNRGBD. “Public” refers to all public datasets that we found to be potentially useful for unstructured road segmentation: BDD100K, Cityscapes, IDD, Mapillary Vistas, Freiburg Forest, RUGD and YCOR. A Cityscapes-pretrained PSPNet-50 (Zhao et al., 2017) is included as a baseline.

From the bottom half of Table 1, it could be concluded that some datasets are not worth combining, as they cause rather poor IoU scores on our own recorded datasets when applied independently. The most extreme case of this is the RUGD dataset, which is likely due to the extremely low viewpoint, driving mostly besides the road instead of on it, and having a limited variety of road types. The top half of Table 1 shows the results of combining multiple datasets to further improve the performance. It shows that removing the lower-performing RUGD data can indeed improve accuracy in some cases, but the impact is minor enough such that simply combining all possible relevant datasets seems to be a reasonable guideline in general.

### 4.2 Path-Finding Qualitative Results

The path planning is difficult to evaluate quantitatively without extensive manual labeling effort, therefore we focus on reporting some failure cases that persist even after using our best segmentation model (trained on Public+NL) and applying all the path-planning improvements described in Sections 3.2.1 and 3.2.2. The NL-val dataset shows worse path-finding failures than the AT-val dataset, hence the examples displayed here are all from NL-val.

Fig. 6 shows four different path-finding failure cases, with blue borders indicating the detected road and the red lines the estimated centerline(s). In (a), a noisy offshoot from the segmentation is sufficiently far away, such that after the BEV transformation it becomes sufficiently large to be recognized as a side road. This is hard to repair afterwards, hence it would likely require a better segmentation, and thus a better trained network, to prevent this. In (b), a tire-track

Table 1: Results of fine-tuning the HRNet-W48 network on several different (combinations of) datasets, where bold scores are highest and italic are second-highest. Top block of the table has been trained on combinations of datasets, middle block on single datasets and bottom block are pre-trained networks from other authors. Each block is sorted by the mean IoU. \*(Note: only road segmentation is evaluated, not any other classes, hence these WildDash scores cannot directly be compared to other works that evaluate on it).

Train Dataset	NL-val	AT-val	WD-val*	Mean
Public+NL	<b>0.928</b>	<b>0.964</b>	0.844	<b>0.912</b>
Public+NL-RUGD	<i>0.927</i>	<i>0.961</i>	0.840	<i>0.909</i>
Public-RUGD	0.907	0.944	0.849	0.900
Public	0.814	0.943	0.818	0.858
YCOR	0.927	0.941	0.784	0.884
IDD	0.875	0.911	<b>0.878</b>	0.888
NL	0.918	0.945	0.593	0.819
Cityscapes	0.886	0.875	<i>0.878</i>	0.880
Mapillary Vistas	0.551	0.805	0.828	0.728
Freiburg Forest	0.652	0.773	0.496	0.640
Rellis3D	0.733	0.733	0.515	0.660
BDD	0.256	0.687	0.829	0.591
RUGD	0.138	0.030	0.371	0.179
<b>Pretrained baselines</b>				
Mseg HRNet-W48	0.395	0.880	0.809	0.695
PSPNet-50	0.609	0.378	0.538	0.508

road is detected as two separate roads. This failure case is common, because all public datasets with tire-track roads also annotate it as separate tracks. Hence, manual relabeling of the public datasets is likely the easiest solution, but time-consuming. In (c), two parallel roads separated by some grass are partially detected as a single road. This is the opposite of the problem in (b): it looks similar to a tire track with grass in the middle, and our private dataset has tire tracks annotated as a single blob, causing this edge-case error. This error only happens in this specific frame and nowhere else in the dataset, hence it is probably not a big issue. In (d), a 4-way crossing results in 5 detected paths. This is rather common when the crossing is still far away, because the shape is not yet well-defined. This problem solves itself once the vehicle more closely approaches the crossing, thus it is not a real problem in practice.

Overall, the failure cases are deemed acceptable for the application of camera control, but application of the proposed algorithm for autonomous driving may require additional robustness improvements, such as temporal filtering.

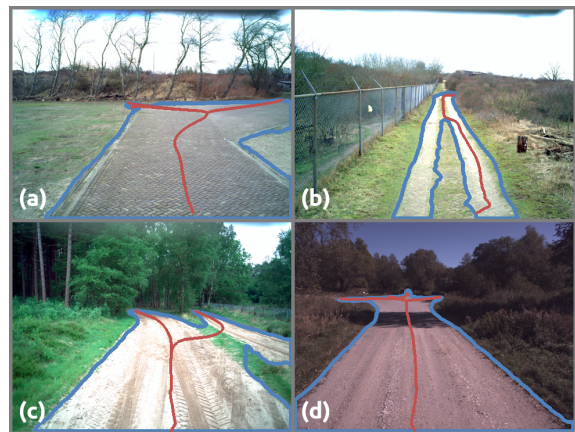


Figure 6: Path-finding failure cases in the NL-val dataset. Road segmentations are outlined in blue, estimated centerlines in red. (a) Noisy segmentation offshoot, (b) separated tire-track road, (c) merged close-together roads, (d) too many paths at intersection (spurious tiny path at top left).

## 5 DISCUSSION

Although the path-planning system works well on our test cases, it also has several limitations. First, while many road types are now present in the combined dataset, there are still edge cases that are not captured. For example, the unpaved road datasets do not contain many images in the dark or with rainy weather, hence performance in those situations is likely to remain limited, as can be partly observed from the WildDash dataset performance.

Second, even though the final performance on our test set is considered good, the performance with only the seven public datasets, thus without the NL images, is moderate at the NL-val set, which shows that generalizing to new road types still remains difficult. Overall, there is still a lack of good public datasets, even small ones, that contain all possible road types.

Third, the path-planning algorithm currently does not take other road users into account. While this is actually beneficial to our specific use case of camera control, it is detrimental when applying this algorithm for autonomous driving. Thus, when trying to use our work for autonomous driving, the most valuable part is the robust road segmentation and not the path-finding algorithm.

## 6 CONCLUSION

In this paper we have proposed a robust path-planning system for any type of road, with a focus on un-



paved roads. There are two main contributions. First, we have combined seven public driving and robotics datasets, which together contain a large variety of road types, and trained a HRNet-W48 network on this data to achieve robust road segmentation. Second, we have developed a path-finding algorithm and improved its robustness to incorrect road segmentation in several ways, allowing for automated control of a vehicle-mounted PTZ camera, which can handle road crossings and forks.

Our experimental results have shown that individual driving datasets contain insufficient variety to allow training of a robust road segmentation system for all road types. Combining seven different public datasets and adding just a small number of semi-automatically labeled images greatly improves the performance to the point that all roads in the test dataset can be accurately segmented. Although this paper is primarily concerned with camera control, the robust road segmentation for a broad class of roads can also be of interest for work in autonomous driving.

Overall, we conclude that robust path planning on any type of road is feasible, but will still require comparable extensive datasets that autonomous driving research has generated for urban environments over the past few years. Until then, combining many existing datasets is a good alternative for generalization.

## REFERENCES

- Cordts, M., Omran, M., Ramos, S., Rehfeld, T., Enzweiler, M., Benenson, R., Franke, U., Roth, S., and Schiele, B. (2016). The Cityscapes Dataset for Semantic Urban Scene Understanding. In *CVPR 2016*.
- Giusti, A., Guzzi, J., Ciresan, D. C., He, F. L., Rodriguez, J. P., Fontana, F., Faessler, M., Forster, C., Schmidhuber, J., Caro, G. D., Scaramuzza, D., and Gambardella, L. M. (2016). A Machine Learning Approach to Visual Perception of Forest Trails for Mobile Robots. *IEEE Robotics and Automation Letters*, 1(2):661–667.
- González, D., Pérez, J., Milanés, V., and Nashashibi, F. (2016). A Review of Motion Planning Techniques for Automated Vehicles. *IEEE Transactions on Intelligent Transportation Systems*, 17(4):1135–1145.
- Grigorescu, S., Trasnea, B., Cocias, T., and Macesanu, G. (2020). A survey of deep learning techniques for autonomous driving. *Journal of Field Robotics*, 37(3):362–386.
- Jiang, P., Osteen, P., Wigness, M., and Saripalli, S. (2020). RELIS-3D dataset: Data, benchmarks and analysis. *arXiv 2011.12954v2*.
- Maturana, D., Chou, P.-W., Uenoyama, M., and Scherer, S. (2018). Real-Time Semantic Mapping for Autonomous Off-Road Navigation. In *Springer Proceedings in Advanced Robotics*, pages 335–350.
- Nakagomi, H., Fuse, Y., Nagata, Y., Hosaka, H., Miyamoto, H., Yokozuka, M., Kamimura, A., Watanabe, H., Tanzawa, T., and Kotani, S. (2021). Forest road surface detection using LiDAR-SLAM and U-Net. *2021 IEEE/SICE International Symposium on System Integration*, pages 727–732.
- Neuhold, G., Ollmann, T., Bulo, S. R., and Kotschieder, P. (2017). The Mapillary Vistas Dataset for Semantic Understanding of Street Scenes. In *ICCV 2017*.
- Paden, B., Čáp, M., Yong, S. Z., Yershov, D., and Frazzoli, E. (2016). A survey of motion planning and control techniques for self-driving urban vehicles. *IEEE Transactions on Intelligent Vehicles*, 1(1):33–55.
- Qureshi, A. H., Simeonov, A., Bency, M. J., and Yip, M. C. (2019). Motion planning networks. In *Proceedings - IEEE International Conference on Robotics and Automation*, volume 2019-May, pages 2118–2124.
- Ronneberger, O., Fischer, P., and Brox, T. (2015). U-Net : Convolutional Networks for Biomedical. In *Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, pages 234–241.
- Shelhamer, E., Long, J., and Darrell, T. (2016). Fully convolutional networks for semantic segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(4):640–651.
- Sun, K., Zhao, Y., Jiang, B., Cheng, T., Xiao, B., Liu, D., Mu, Y., Wang, X., Liu, W., and Wang, J. (2019). High-resolution representations for labeling pixels and regions. *arXiv 1904.04514v1*.
- Valada, A., Oliveira, G. L., Brox, T., and Burgard, W. (2016). Deep Multispectral Semantic Scene Understanding of Forested Environments Using Multimodal Fusion. In *International Symposium on Experimental Robotics (ISER)*.
- Varma, G., Subramanian, A., Nboodiri, A., and Chandraker, M. (2019). IDD : A Dataset for Exploring Problems of Autonomous Navigation in Unconstrained Environments. In *IEEE Winter Conference on Applications of Computer Vision (WACV)*.
- Wigness, M., Eum, S., Rogers, J. G., Han, D., and Kwon, H. (2019). A RUGD Dataset for Autonomous Navigation and Visual Perception in Unstructured Outdoor Environments. *IEEE International Conference on Intelligent Robots and Systems*, pages 5000–5007.
- Yadav, S., Patra, S., Arora, C., and Banerjee, S. (2017). Deep CNN with Color Lines Model for Unmarked Road Segmentation. In *IEEE International Conference on Image Processing (ICIP)*, pages 585–589.
- Yu, F., Chen, H., Wang, X., Xian, W., Chen, Y., Liu, F., Madhavan, V., and Darrell, T. (2020). BDD100K: A Diverse Driving Dataset for Heterogeneous Multitask Learning. In *CVPR 2020*, pages 2633–2642.
- Zendel, O., Honauer, K., Murschitz, M., Steininger, D., and Domínguez, G. F. (2018). WildDash - Creating Hazard-Aware Benchmarks. In *Proceedings of ECCV*, pages 407–421.
- Zhao, H., Shi, J., Qi, X., Wang, X., and Jia, J. (2017). Pyramid Scene Parsing Network. In *CVPR*.