STI - Microtechnique - Master V

Master Project - performed at the VITA laboratory

Monocular 3D Vehicle Detection with Self Attention Mechanisms

Task: Study on how self-attention mechanism can be used for Monocular 3D vehicle detection.

Author :
Maxime Bonnesoeur

Supervisors :
Prof. Alahi Alexandre
Bertoni Lorenzo

External expert :
Dr. Bagnato Luigi

January 15, 2021

Contents

1 Introduction **2**

2 Related works **3**

3 Approach **5**

4 Vehicle 3D pose localization **5**

4.1 Base implementation 6

4.2 Attention-based Network 6

4.3 Experiment 8

5 Scene refinement **9**

5.1 Input formatting 9

5.2 Network architecture 10

5.3 Experiment 12

6 3D key points extrapolation **12**

6.1 Extraction of the Depth component 12

6.2 Network 13

7 Conclusion **15**

A Apolloscape **16**

B LSTM implementation **16**

References **18**

Abstract

Monocular 3D object localization is an essential field of study for autonomous driving. It is nonetheless a challenging task since some samples are occluded. In this current field, most detector consider each instances as independent and do not consider their joined relationships. We introduce a novel monocular 3D vehicle detector using attention-based mechanisms.The objective being to take the relation between instances into account. Experiments demonstrate that our method yields competitive results in the KITTI 3D[1] detection benchmark and is the first to propose an self-attention approach at the scene level. We also develop a technique on the recent Apolloscape[2] dataset to extract the 3D meshes of the visible vehicles.

1 Introduction

In today's autonomous driving industry, one of the main challenge is to estimate other vehicle's localization in the 3D space. This is mainly performed by the use of a Lidar or with a redundancy of sensors. However, those are either costly or require a precise calibration to work efficiently.

In this work, we propose a monocular 3D vehicle localization method such as the one presented in the Figure 1. The aim is to be able to reason on each vehicles instances as well as on the overall scene to provide the estimation.

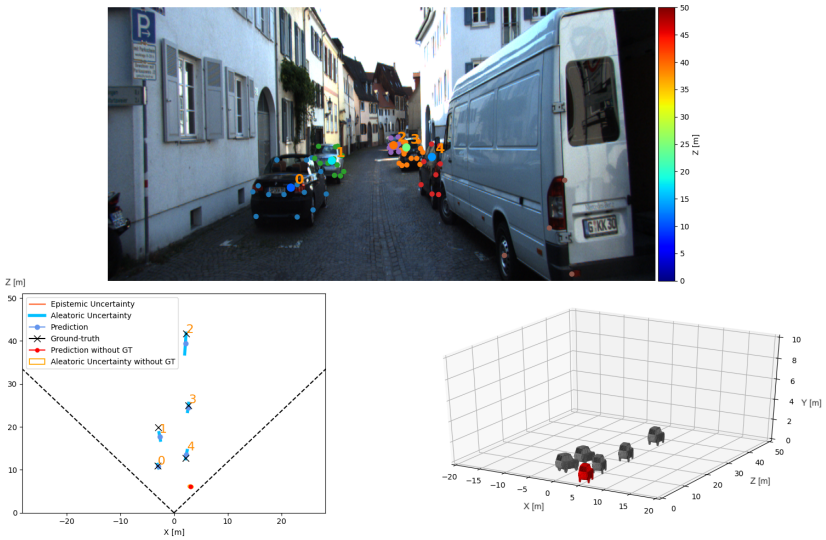

Figure 1: Vehicle Monocular 3D localization (Grey vehicle: vehicle with ground truth, Red vehicle: vehicle detected without ground truth)

Our project leverages on a pose detector to identify the vehicles[3] and a localization module[4] tuned for our task. A self-attention mechanism is then used to refine our estimates based on the configuration of the scene. To the best of our knowledge, this is the first time that a

self-attention mechanism is used to extract dependencies between instances at the scene level.

Indeed, as in the Monopair[5] work, we believe that taking into consideration the interactions between the instances in a scene is primordial to obtain accurate position estimations despite occlusion. However, our guess is that those interactions can be deduced by the network and do not require to be computed beforehand.

Additionally, in the course of this project, we developed a method to extract the 3D shapes of vehicles based on the Apolloscape[2] dataset. This feature is allowing us to gather additional information on the environment.

2 Related works

In this section, we will review some other methods on monocular object detections and related works that made this project possible.

Monocular 3d Object detection

Given the limited 3D information of monocular images compared with Lidar or stereo vision, prior knowledge and auxiliary information are widely used for 3D object detection. Mono3D[6] focuses on the hypotheses that 3D objects are on the ground plane. Prior 3D shapes of objects can also be leveraged to reconstruct the bounding box. For example, Deep MANTA [7] predicts 3D vehicles localization utilizing key points and 3D CAD models of cars fitted on the bounding boxes.

Some algorithms also consist of only the RGB image as input without relying on external parameters. MonoGRNet[8] predicts 3D object localization from a monocular RGB image considering geometric reasoning in 2D projection and the unobserved depth dimension. It aligns itself in the trend of predicting multiple factors (namely depth, 2D box detection, edge detection) that are blended to obtain a final localization. MonoDIS[9] on the other hand leverages a disentangling transformation for 2D and 3D detection losses, this technique allows a simplification in the training dynamics.

All the object detectors mentioned above focus on predicting each individual object from the image. The spatial relationship among objects is not considered.

Geometric constraints enforcement

Some other publications such as MonoPair[5] are using environmental constraint to refine their localization. They focused on guiding the 3D object localization by adding spacial constraints to respect for the vehicles placement. In this case, they proposed to jointly compute the object localization as well as a spatial constraint matched between object pairs. This constraint is modeled as a key point located in the geometric center between the 2 neighboring objects. By enforcing this constraint, Monopair was able to achieve state of the art result despite the simplicity of its network.

Transformer for the computer vision field

At the time of writing of this work, an interesting take on computer vision is gathering interest. Thanks to the recent advancement made in the Neural Language processing field, new technique such as self-attention mechanisms introduced in the paper "Attention is all you need"[10] are being used in vision-based tasks. This is notably the case for the paper "An Image is Worth 16x16 Words"[11] where self-attention mechanism are being leveraged for instance recognition tasks while producing state-of-the-art results.

Human pose estimator - Openpifpaf

Openpifpaf is an algortihm that uses a Part Intensity Field (PIF) to localize body parts and a Part Association Field (PAF) to associate body parts with each other to form full human poses. Developped at the Vita laboratory, its architecture is based on a fully convolutional, single-shot, box-free design. It was trained on the COCO key point dataset[12] which contains the positions of the body parts of a human being on an image.

The algorithm was modified by the VITA laboratory during the course of this project to detect the key points of the corners of the vehicles. Even though this process was not implemented during this particular thesis, it is interesting to see how it was trained, and what does it output. It was trained on the Apolloscape[2] dataset which is thoroughly described in the Appendix section A.

To train the openpifpaf model, a set of 24 key points sampled from the 66 available ones from the Apolloscape dataset were selected. These key points are the joints of a "skeleton" which is then trained to fit the vehicles in the images.

The training resulted in an average precision of 70.0 % is reached for the instance detection. Also, the skeleton seems to be fairly well fitted on the vehicles as the Figure 3 shows.

Figure 2: Original image before processing

Figure 3: Scene processed with openpifpaf

Nonetheless, a throughout comparison with other techniques is not possible since the 2D pose recognition for cars on Apolloscape is a new field of study. Also, this technique only displays the visible keypoints from the 2D images.

3D pedestrian localization - Monoloco

Monoloco[4] is a position estimator for human. It takes as an input the 2D poses of human beings and outputs their estimated 3D position and the corresponding confidence interval of the estimation. The algorithm was also developped at the VITA laboratory

3 Approach

We are using a two step approach in our technique. The first step identifies the vehicles and predict a set of 2D poses for each instances. Those key points represent the corners of a vehicle and are often incomplete due to severe occlusion or self-occlusion.

The second step consist in using those 2D pose sets re-projected in the image plane to estimate the 3D pose of the vehicles.

Our approach is summarized in the figure 4.

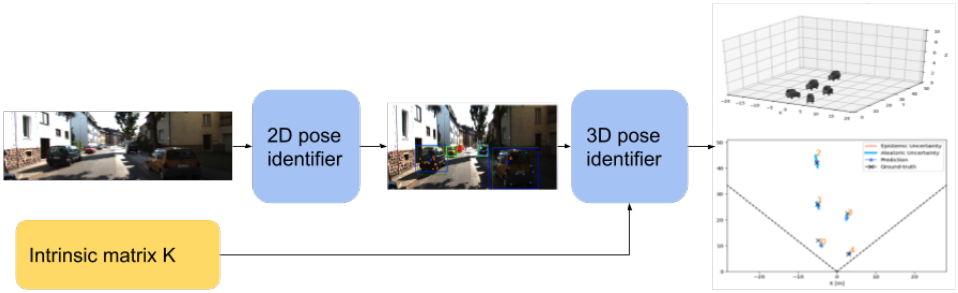

Figure 4: Structure of the algorithm

The 2D car poses identifier is the modified Openpfipaf[3] presented in the Related Work Section 2 and the 3D pose identifier is a modified version of Monoloco[4] that accommodates new self-attention mechanisms on the scene and instance levels.
The fact that both these algorithms are developed at the VITA laboratory is no coincidence. First and foremost, both networks are attaining top of the line results in their own respective fields. Also, the ability to work closely with the ones that created those algorithm allowed a deeper understanding of the subtleties of each of them.

4 Vehicle 3D pose localization

For the next step of our project, we are using the KITTI[1] dataset for our localization task. The reason for using a different algorithm here is due to the freshness of Apolloscape.
Historically, the task of instance localization has been already well studied on KITTI and we have actual material to estimate the performance of the algorithm.

Firstly, the regular Monoloco framework was adapted for vehicle localization. Then, a self-attention network is developped as an alternative from the Monoloco backbone. Finally, a refinement step taking into account the scene disposition was added.

4.1 Base implementation

As the input of our network, the 24 key points obtained from openpifpaf are processed before being fed to Monoloco. Each key point are described as follow : $kps[i] = (x'_i, y'_i, c_i)$ with $i \in (0, N-1)$. where i is the index of the key point and N being the number of key points. x',y' and c are respectively the X and Y coordinate in the pixel frame and the confidence with which this key point was detected. Since we want our algorithm to be generalized to other camera parameters, the X and Y coordinates are projected beforehand on the image frame thanks to the intrinsic camera matrix K.

The newly obtained x and y coordinate are the used in the flattened vector of length $3 * 24 = 72$ fed to the network described in the Figure 6. The loss part is identical to the one described in the Monoloco[4] paper. The parameters d and b describing the depth and the localization uncertainty are trained with a Laplacian loss. The other joined parameters namely $x, y, width, height, length$, *relative orientation*, *absolute orientation* are trained using a L_1 loss.

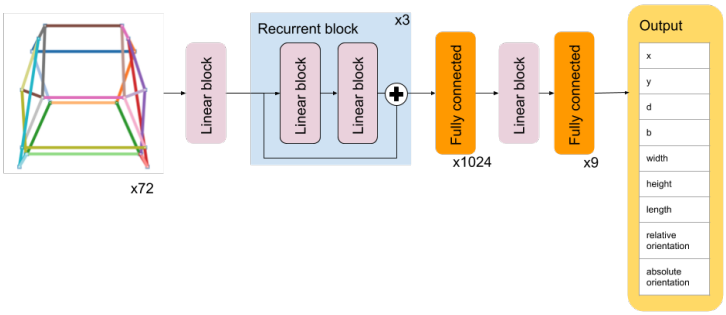

Figure 5: Base network for 3d pose estimation

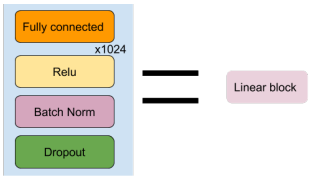

Figure 6: Linear Block implementation

Originally, Monoloco uses the full body pose to provide its depth estimation. In our case however, the vehicle are often self occluded or severely occluded by their environment.

Concerning the occluded key points, those are denoted with a confidence $c_i = 0$ and are replaced replaced by the mean (in the x and y dimension respectively) of the non-occluded key points of each instance.

4.2 Attention-based Network

The attention-based strategy originated from the idea of removing the constraint of occluded keypoints thanks to a self-attention mechanism. Indeed, in first publication on self-attention[10] , the task was to perform sentence translation for varying sentences length. In our case however, we want to use the attention mechanism to estimate the dependencies between the keypoints and thus reduce the occluded keypoints impact.

To do so, the inputs, are arranged in a different fashion described in the Figure 7. For the ones used to NLP formatting, the array of keypoints can be assimilated to a sentence and each keypoint coordinates to a word. Each key point is composed by its x and y coordinate, its confidence as well as a positional embedding which consist in a set of a cosinus and a sinus.

The objective here is to have an unique identifier for each key points in the set of size N=24.

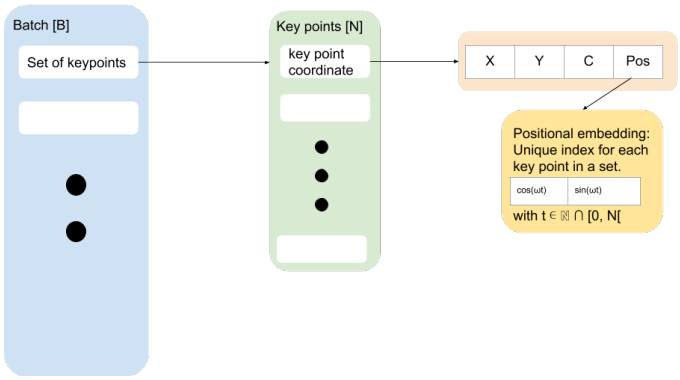

Figure 7: Embedding of the key points for the attention based network

Regarding the occluded key points, those are replaced with a pad as described in the Figure 8. This pad is an array of 0 not containing any positionnal embedding. The position of the pads is also transmitted to the multi-headed attention as a mask to remove their contribution during the training and compute the dependecies between the key points.

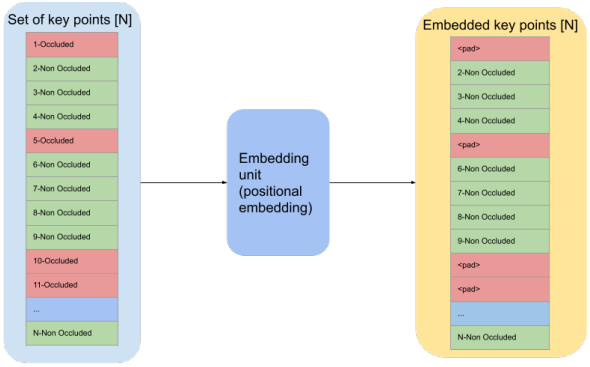

Figure 8: Occluded key points handling

The resulting network is taking the form of multiple layers of self attention and feed forward passes presented in the Figure 9. The three arrows entering the multi-headed attention block represent the key, query and value vector inputs that the attention mechanism is using for its processing. For more information on the processing step, please refer to Vaswani's paper[10].

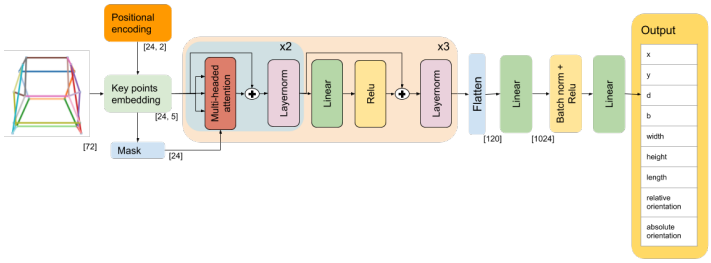

Figure 9: Self-attention network for vehicle localization

The multi-headed attention module is presented in the Figure 10. This module aims is to compute different independent attention processing to gather a wider diversity of relationship between the key points sets.

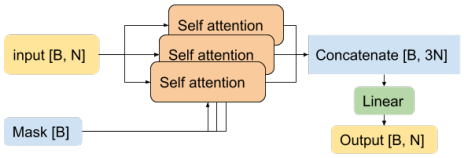

Figure 10: Multi-head attention module

4.3 Experiment

	Average Localization Accuracy [ALA](%) ↑			Absolute Linear Error[ALE] (m) ↑ [Recall (%)]			
Models	<0.5 m	<1m	<2m	Easy	Medium	Hard	All
Attention network	21.21	33.52	45.89	1.05 [98%]	2.06 [81%]	**1.66** [37%]	**1.82** [71%]
Monoloco - mean	**22.68**	**36.43**	**52.05**	**0.98 [98%]**	**1.98 [89%]**	2.11 **[43%]**	1.86 **[75%]**
LSTM	3.138	6.129	1.03	3.23 [75%]	4.17 [26%]	3.04 [9%]	3.57 [28%]

Table 1: Results and comparison for vehicle localization

With the newly obtained technique, we are not reaching a higher milestone compared to the padded Monoloco implementationas displayed in the Table 1. The results are too similar while the recall from the previous method remains superior.
To see if our implementation was faulty, we also tried to replace the self-attention layers with LSTM ones. The detailed implementation is presented in the Appendix B.
Thus, this network may be considered as an alternative to an already existing technique but not as an overall improvement over the original Monoloco network. The results show a worse performance of the LSTM compared to self-attention mechanisms. The numbers are nonetheless logical since the LSTM only take into account the relationship between neighborhing key points in an array. In our case however, we want to look at the relationship between every key points.
Thanks to this analysis, and the closeness of the results between a padded Monoloco and the self-attention mechanism, we are defining the hypothesis that there is only so much 3D information that can be gathered from these partial sets of keypoints. Thus to improve our

results, additional information needs to be gathered.

We briefly show the results obtained with our modified Monoloco technique in the Figure 11. It is clear that the localization technique is working in this scenario. However, uncertainty and errors are remaining as can be seen in the bottom right map.

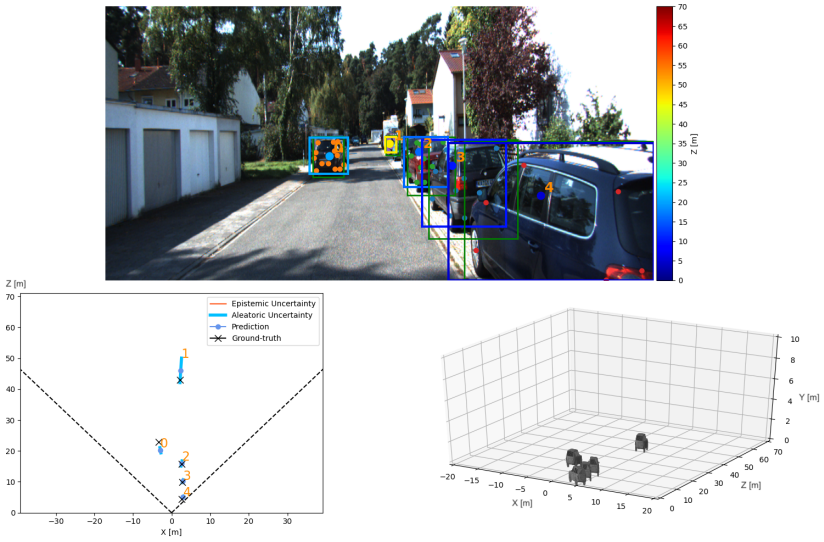

Figure 11: Vehicle localization from 2d pose

To take the project further, an other approach was envisaged. In order to gather more information from our keypoints set, a refinement step at the scene level was implemented to gather additional scene-level features.

5 Scene refinement

Our approach to improve our previous results was to look at the entire scene to perform a refinement step. Indeed, as the Monopair[5] publication stated, the relation between instances in a scene should not be underestimated for refining the final localization.

It is especially true with vehicles where a lot of occlusion can happen in dense traffic for example. In such scenarios, the loss of key points for vehicles is becoming a critical factor for our kind of algorithm.

One problem that we are faced with however would be that the number of vehicles in a scene is a varying factor. Such situation puts us in the same scenario as the one in the Section 4.2. For this reason, an attention-based mechanism is here again used to perform the refining step.

5.1 Input formatting

To understand how this attention mechanism will work, we need to take a look at how the inputs will be formatted.

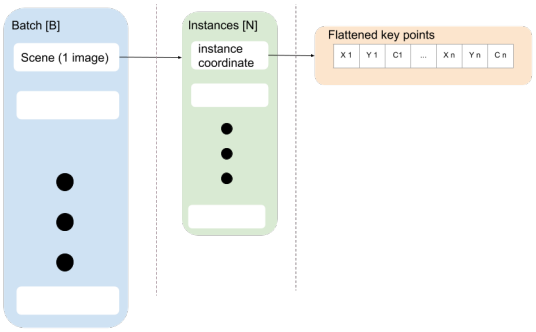

Figure 12: Representation of the input formatting

As displayed in the figure 12, we have several levels to consider with this formatting. In the previous section 4, the model was only looking at each instances individually. In the figure 12, it would be as if the network was looking at a batch of size $B * N$ of inputs which are in this case the flattened key points.

However, if we want to look at the interaction between the instances, we need to look at the entire scene. Our dataset will thus be divided for this sequence in a set of B scenes each containing N instances. For those used with the transformer and NLP in general, we can see the instances as sentences and the flattened key points as the embedding for the words.
The number of instances is a fixed number (in our case, N = 20, the maximum number of vehicle in a scene), if a scene contains less than 20 vehicles, the rest of the inputs are replaced with a 0 padding without positional embedding. Those occluded inputs are also given to the attention mechanism as a mask. This will thus remove their contribution from the training of the attention-based mechanism.
The rest of the instances in a scenes are then ordered beforehand in the order of apparition of their mean position (key point wise) alongside the X axis.

5.2 Network architecture

As can be seen on the figure 13, we are reasoning in two parts in our model.

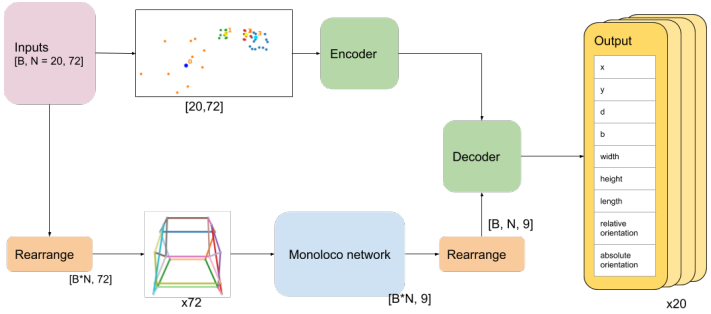

Figure 13: Architecture of the network for the scene-level refinement

At first, the inputs are rearranged to be processed by our original monoloco network. The

objective is is to keep the strength of the instance-based 3D pose estimator before entering the refinement step. The output is then rearranged to keep the scene arrangement for the decoding step.

At the same time, an encoder using attention mechanism will process the instances by drawing relationship between them at the scene level. The mechanism is described in the figure 14. The mask extracted from the embedding is the position of the padded inputs for the scene formatting. This way, the network only takes into consideration the available scenes during its processing.

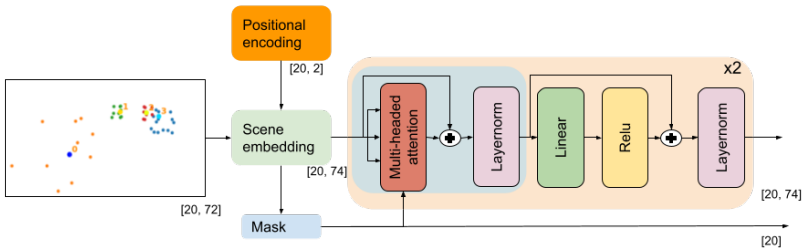

Figure 14: Encoder for the refining step

Both of these scene level and instance level processes are then fed into our decoder that will link the two reasoning together and provide a refined outputs. The process of the decoder is represented in the figure 15. The network uses the encoder outputs as his key and value input for its second self-attention mechanism of each layer which links the embedding found in the encoding step to the decoder's inputs. Additionally, one can notice the presence of a mask as an input. This mask comes from the previous encoding step to ensure that the network is not training on the padded key points which may be unidentifiable after the instance-based processing.

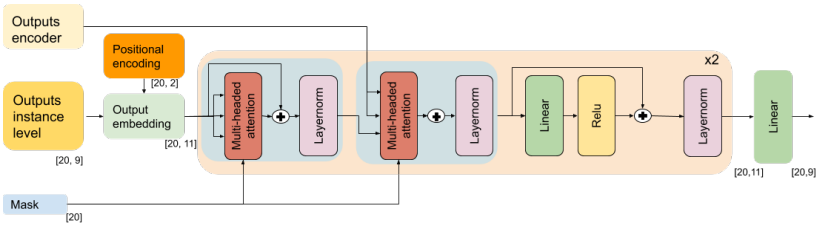

Figure 15: Decoder for the refining step

The loss is then back-propagated only for the non-padded instances for the scenes. The same losses as in the section 4 are used.

5.3 Experiment

	Average Localization Accuracy [ALA](%)			Absolute Linear Error[ALE] (m) [Recall (%)]			
Models	<0.5 m	<1m	<2m	Easy	Medium	Hard	ALL
Scene refinement network	20.67	34.33	50.53	1.02 [98%]	2.06 [88%]	2.30 [39%]	1.96 [74%]
Scene refinement network	20.15	34.12	48.88	1.06 [98%]	2.05 [86%]	2.09 [34%]	1.96 [71%]
Monoloco - mean	22.68	36.43	52.05	0.98 [98%]	1.98 [89%]	2.11 [43%]	1.86 [75%]
Attention network	21.21	33.52	45.89	1.05 [98%]	2.06 [81%]	1.66 [37%]	1.82 [71%]
M3D	12.62	23.78	39.68	2.02 [99%]	3.53 [90%]	2.82 [66%]	3.02 [72%]
Monodis	28.66	43.91	58.27	0.72 [99%]	1.43 **[91 %]**	**1.31** [66%]	**1.21** [72 %]
MonoGrnet	**30.23**	**46.44**	**61.04**	**0.66 [99 %]**	**1.36** [88%]	1.72 [44%]	1.34 **[77 %]**
3dop (stereo)	27.01	41.73	55.3	0.66 [98%]	1.81 [91%]	1.68 **[70%]**	1.58 [72%]

Table 2: Results and comparison for vehicle localization and scene refinement

At the end of our experiment, the results obtained by the scene refinement step are not conclusive. They are still not at the same level as the other algorithms that we proposed earlier in this study in the Table 2. Our performances are also not competitive enough compared to state of the art techniques such as Monodis [9] or MonoGrnet[8].

The main learning that we can get from this experiment is that the current refinement process seems only seems to add complexity to our problem. This is apparent on the table by having worse results, by a small margin, compared to the original method that we wanted to improve.

It seems that the current encoder is not able to create some relationships between the sets of keypoints in a scene. A future work task would be to see if a different and simpler embedding for the encoder part would allow it to draw relevant constraints between the instances.

6 3D key points extrapolation

Additionally to our work on self-attention for vehicle localization, we performed during this project, predicting the depth component of each 2D keypoint that we have available. This is allowing us to obtain additional information concerning the shapes of the instances that we want to estimate and is a field that has been gathering interest recently.
This work was preformed on the Apolloscape[2] since it contains the 3D cad model and precise positioning of its vehicle instances. More information on this dataset are presented in the Appendix A.

6.1 Extraction of the Depth component

The 3D meshes are first projected into the 3D space thanks to the available CAD models and 3D positioning available. Those "surfaces" in the 3D space will help us to extract the depth component of our predicted 2d key points obtained with the modified openpifpaf network.

First and foremost, the vertices of the 3D meshes are re-projected onto the 2D space as shown in the Figure 16.

Figure 16: Projection of the 3D vertices (grey key points) onto the image plane. The red key points are the one obtained with openpifpaf

Then, the 30 nearest projected vertices from each key point of openpifpaf are gathered. We are selecting this subset of vertices since we do not want the closest re-projected key point to give us its depth. In fact, if we used this technique, a vertice that is occluded in the 3D space will have the same amount of chance to give its depth than the visible vertices.

For this reason, we take the smallest depth from the 30 selected vertices since the visible part of each vehicle has an inferior depth compared to its occluded side.

The results of the re-projection back to the 3D space in the Figure 17 demonstrates that the obtained fitting is close enough to the reality. Hence, we can start training our network.

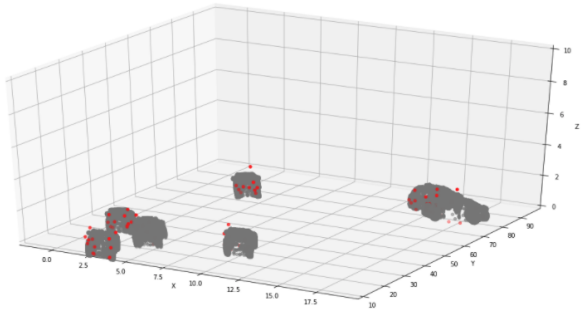

Figure 17: 3D set of vertices (grey key points) and the associated estimated key points (red key points) re-projected in the 3D space

6.2 Network

For this step, we only took the modified Monoloco network that was proved to be the most reliable one for our task and add to its output the required N=24 depth components for each key points. This is represented on the Figure 18 . In this case, the z_1 to z_{N-1} parameters are trained using a L_1 loss. Of course, only the loss for the non-occluded key points is being back-propagated through the network.

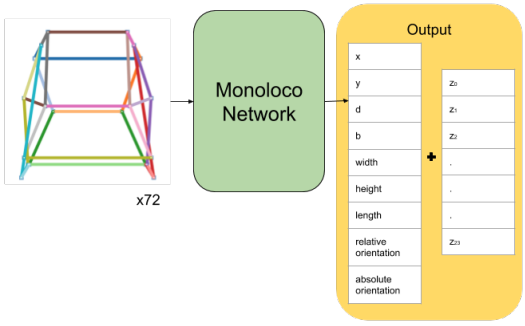

Figure 18: Network used for the 3D key points depth retrieval

6.3 Experiment

The task of depth retrieval for the key points in the apolloscape dataset was, at the time of this writing and to the best of our knowledge not documented until now. We can nonetheless proceed to a qualitative analysis of our results.

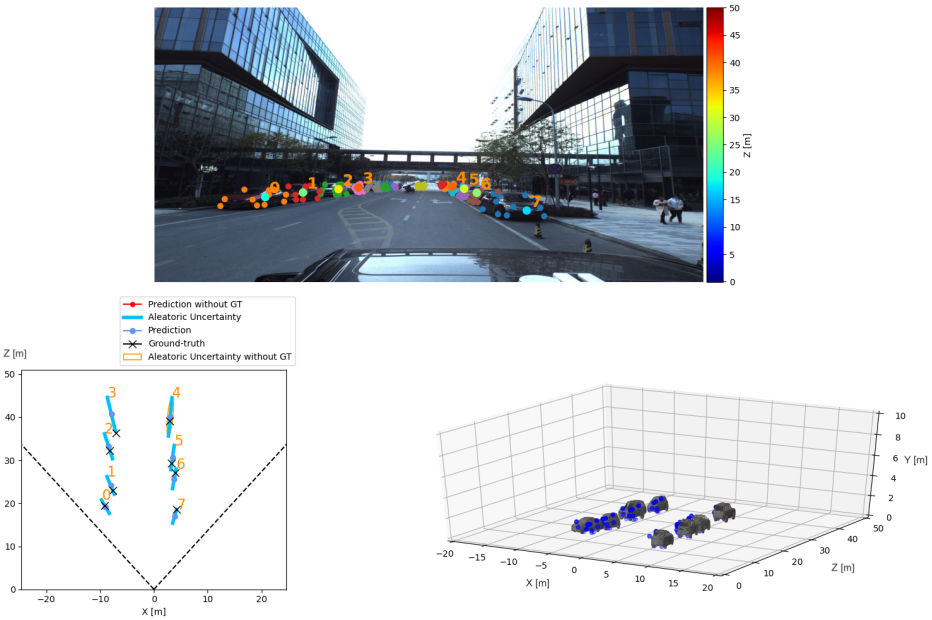

Figure 19: Position estimation and 3D key points prediction (Predicted key points clouds in blue on the bottom right graph)

On the Figure 19 , we can see that the depth component for the keypoints seems to be coherent with the actual positioning of the vehicles in the 3D space. The keypoints are positioned accurately around the car which is giving us some additional information about their shape.

Even though this technique was not leveraged upon during this project, we believe that it might prove itself useful for 3D shape reconstruction or vehicle identification tasks in future works.

7 Conclusion

In summary, a new 3D pose estimation leveraging on 2D pose for vehicles was implemented. The new technique leverages on self-attention mechanism to tackle the ill-problem of occlusion in the 2D pose. Furthermore, a refinement step harvesting information at the scene level was also developed using self-attention.
Even though those implementations did not bear state-of-the-art results, we believe that the way of reasoning at the instance and scene level at the same time is still a valid argument.

A relevant future work to improve upon those results would be to think about a new kind of embedding that would make it easier to draw connection and deduce constraints between the vehicles in a scene.

Additionally, a key point depth estimator was implemented to retrieve the 3D shape of non-occluded part of vehicles. This work could be a relevant starting point to develop complete 3D shape reconstruction based on partial vehicle shape information.

Signature :

A Apolloscape

A new dataset called ApolloCar3D [2] published in 2019 is then envisioned to train our model. It is a high resolution dataset with many images, a large diversity of environments. Furthermore, it contains the 3D CAD models of each vehicle with their accurate positions. The specificities of the dataset are the following:

- Number of frames : 5'277 frames
- Resolution : 3384 x 2710
- Number of referenced car instances : $>$ 60'000
- referenced car models (3D cad models) : 79

Assessment of the dataset

Apolloscape possess the exact positioning of the vehicle instance with 79 different CAD model to fit the real-life vehicles with their 3D counterparts.

Furthermore, a set of 2D key points is also manually annotated on top of the 3D meshes of the apolloscape dataset. They are a total of 66 different key points which are only visible ones. Those key points do not possess any third dimension and are depicted in the figure 20

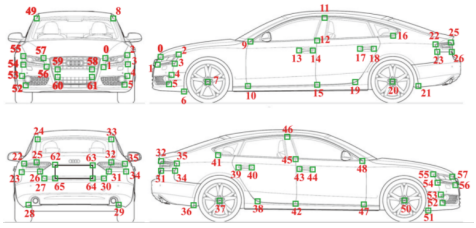

Figure 20: Correspondence of the set of 2D key points on a vehicle of type sedan

The reason why we have both the meshes of the 3D CAD model and those 2D key points is that the vertices of each CAD model are assigned to a different position on the vehicle. For example, the vertices 0 on the CAD model of a sedan can correspond to a wheel while the vertices 0 of a SUV may correspond to a rear light.
For this reason, Openpifpaf in this work was trained on this subset of 66 key points.

B LSTM implementation

In the Figure 21, the LSTM model used to benchmark our attention-based transformer is presented. The same formatting for the key points as with the self-attention mechanisms displayed in the figure 7 is used for the LSTM implementation.

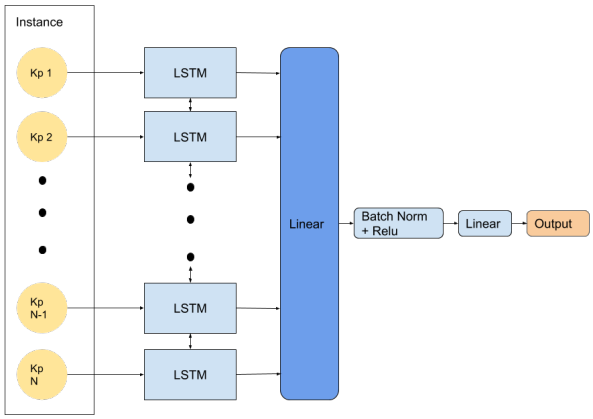

Figure 21: LSTM network implementation for vehicle localization

References

[1] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision meets robotics: The kitti dataset. *International Journal of Robotics Research (IJRR)*, 2013.

[2] Xibin Song, Peng Wang, Dingfu Zhou, Rui Zhu, Chenye Guan, Yuchao Dai, Hao Su, Hongdong Li, and Ruigang Yang. Apollocar3d: A large 3d car instance understanding benchmark for autonomous driving. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun 2019.

[3] Sven Kreiss, Lorenzo Bertoni, and Alexandre Alahi. Pifpaf: Composite fields for human pose estimation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[4] Lorenzo Bertoni, Sven Kreiss, and Alexandre Alahi. Monoloco: Monocular 3d pedestrian localization and uncertainty estimation. *The IEEE Conference on Computer Vision and Pattern Recognition (ICCV)*, 2019.

[5] Yongjian Chen, Lei Tai, Kai Sun, and Mingyang Li. Monopair: Monocular 3d object detection using pairwise spatial relationships, 2020.

[6] Tong He and Stefano Soatto. Mono3d++: Monocular 3d vehicle detection with two-scale 3d hypotheses and task priors, 2019.

[7] Florian Chabot, Mohamed Chaouch, Jaonary Rabarisoa, Celine Teuliere, and Thierry Chateau. Deep manta: A coarse-to-fine many-task network for joint 2d and 3d vehicle analysis from monocular image. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jul 2017.

[8] Zengyi Qin, Jinglu Wang, and Yan Lu. Monogrnet: A geometric reasoning network for monocular 3d object localization, 2020.

[9] Andrea Simonelli, Samuel Rota Rota Bulò, Lorenzo Porzi, Manuel López-Antequera, and Peter Kontschieder. Disentangling monocular 3d object detection, 2019.

[10] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2017.

[11] Bichen Wu, Chenfeng Xu, Xiaoliang Dai, Alvin Wan, Peizhao Zhang, Zhicheng Yan, Masayoshi Tomizuka, Joseph Gonzalez, Kurt Keutzer, and Peter Vajda. Visual transformers: Token-based image representation and processing for computer vision, 2020.

[12] Tsung-Yi Lin, Michael Maire, Serge Belongie, Lubomir Bourdev, Ross Girshick, James Hays, Pietro Perona, Deva Ramanan, C. Lawrence Zitnick, and Piotr Dollár. Microsoft coco: Common objects in context, 2014.