Self-Supervised Camera Pose Estimation With Geometric Consistency

Anirudh Chakravarthy*
achakrav@andrew.cmu.edu

Vanshaj Chowdhary*
vanshajc@andrew.cmu.edu

1 Introduction

Camera pose estimation is critical for many downstream tasks such as 3D reconstruction, SLAM, novel view synthesis, etc. Typically, to estimate camera poses from multiple views, structure from motion techniques such as COLMAP [1] are widely in practice, due to well-tuned parameters and strong real-world generalization. However, COLMAP is an offline method and requires large computational resources post-capture to estimate the relative poses. Furthermore, it requires several images from different viewpoints, each having significant overlap with some other image, which may not be available or possible in practice.

To address these concerns, there has been a renewed attempt to solve camera pose estimation using learning-based methods. Recently, RelPose [2] attempted to solve this problem by modeling epipolar geometry using vision transformers and a supervised loss based on the ground-truth camera poses. While these methods aim to run in an online manner for applications such as AR/VR, these methods require access to ground-truth poses as supervision. This may not always be possible, since odometry equipment is expensive or the images may be casual/internet images (with no known ground-truth poses).

In order to address these issues, in this work, we aim to look at performing camera pose estimation in a self-supervised manner. To this end, we leverage advances in self-supervised monocular depth. Specifically, using the Monodepth2 [3] framework, we aim to supervise a pose estimation network, in order to solve this problem. However, we note that Monodepth2 already has a CNN-based pose estimation network that works well in practice. Therefore, through this project, we aim to examine whether this self-supervised framework which has been shown to work well for CNN-based architectures, can also generalize to transformer-based architectures such as RelPose.

To summarize, our contributions are two-fold:

1. We train a vision transformer, namely RelPose, in a self-supervised manner. We investigate it's performance and provide detailed insights and analysis.
2. We extend RelPose to outdoor and dynamic scenes. The authors used this framework on static environments, however, we aim to break this assumption and evaluate it's performance on autonomous navigation environments to investigate it's effectiveness in real-world scenarios.

Our code is made available at `https://github.com/vanshajc/self-supervised-relpose`

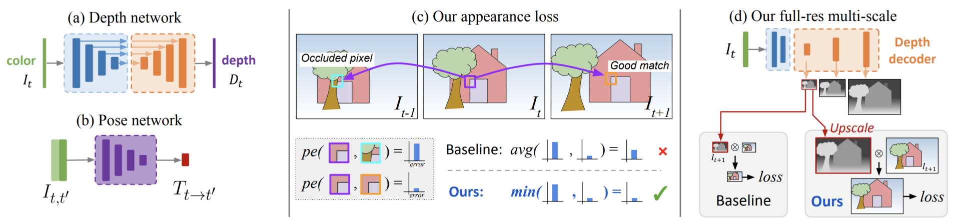


Figure 1: Monodepth2 [3]. Self-supervised network that trains both depth and pose simultaneously using re-projection and similarity based losses in a self-supervised manner.

2 Related Work

2.1 Learning-based Epipolar Geometry

Classical techniques like 8-point algorithm involve finding a set of point correspondences $\{p_i, p_i'\}$ and optimizing for the fundamental matrix F such that $p_i' F p_i = 0$. This is done by creating the matrix A composed of the Kronecker product of the corresponding homogeneous points $\{p_i, p_i'\}$. Given the matrix A, the fundamental matrix is simply the corresponding eigenvector of the smallest eigenvalue of $A^T A$. Lastly, we enforce that the fundamental matrix is rank-deficient. The essential matrix can also be formulated using a similar approach given known camera intrinsics. We wish to impose this epipolar constraint into the pose estimation network as well.

RelPose [2] is a state-of-the-art method for estimating the relative camera rotation and translation between two views. Given two views, we divide them into patches and use a vision transformer to feed into an essential matrix module. This module uses the 8-point algorithm formulation to construct an essential matrix, and thereby, the relative rotations and translations.

We identify two key assumptions which prevent RelPose from generalizing to the real-world:

1. Assumes there are no dynamic objects in the training data
2. Assumes access to ground truth relative poses

We wish to relax these assumptions and investigate whether RelPose still works well under unconstrained and dynamic environments.

2.2 Self Supervised Monocular Depth and Pose Estimation

Monodepth2 [3] is a state-of-the-art network which uses self-supervised losses for monocular depth estimation. It consists of two networks, a depth network (Fig. 1a) and a pose prediction network (Fig. 1b). Given a sequence of images, the depth network individually predicts the depth of the pixels in each image. The pose prediction network, given two pairs of images from the given sequence, aims to predict the relative rotation and translations between the cameras (intrinsics are the same) using a CNN-based architecture. The depth network is supervised using a reprojection losses after applying the predicted pose to the source image I_t. Given the poses of the cameras and the predicted depths, the reference images are warped to the source image frame. Then, the reprojection loss is simply a L1 loss on the depth map in addition to a SSIM loss. In our work, we use Monodepth2 as the self-supervision framework.

3 Our Approach

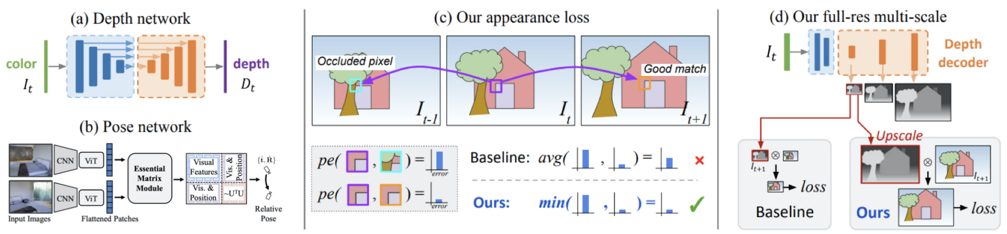


Figure 2: Our Architecture. Our proposed network architecture. In contrast to [3], we use RelPose to predict poses.

3.1 Overview

In our project, we aim to train a transformer-based pose estimation network (RelPose) under a self-supervised framework. To this end, we rely on advances in monocular depth estimation (specifically, Monodepth2). Monodepth2 uses a CNN-based pose estimation network, which, given two images, predicts the relative transformation between the two images. Monodepth2 implicitly supervises the pose prediction network using reprojection losses, while also detecting moving objects using an auto-masking strategy. The authors noted that this was not a difficult problem to solve, since it does not require a particularly deep pose prediction network. This suggests that the self-supervised framework is sufficient to solve pose estimation.

Therefore, in theory, a transformer-based architecture should also be able to recover the relative pose given a pair of images. In order to validate this hypothesis, we replace the pose prediction network from Monodepth (Fig 1b) with RelPose. Our proposed method is illustrated in Fig 2, where we replace the pose prediction network with RelPose.

3.2 Implementation Details

We follow the exact training recipe provided in Monodepth2. We use the KITTI Odometry benchmark [4] to train and evaluate our experiments. The training set consists of 8 sequences while the validation set consists of two sequences (9 and 10). We use the validation sequences to report our results.

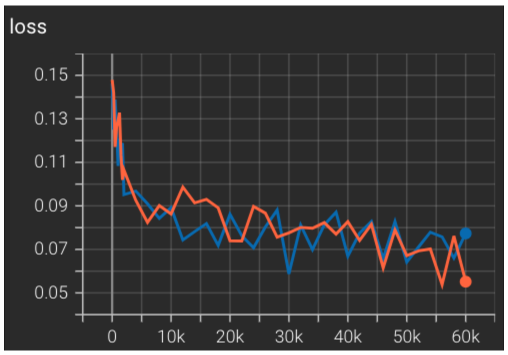
We use the average trajectory error metric mainly to compare our methods with the baseline. Average trajectory error takes ground truth poses and predicted poses over short segments of the trajectory and computes the average error over all the segments. Other evaluation metrics such as translational and rotational errors could also be measured, but were skipped in the interest of time.

4 Experiments

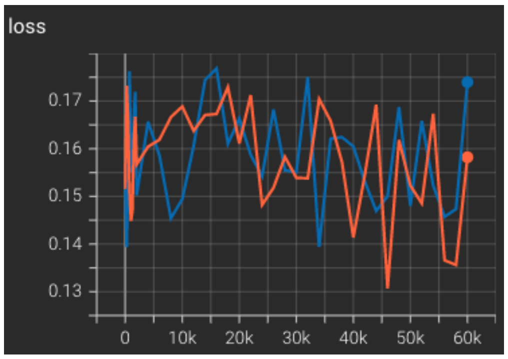
Method	Supervision	ATE (Seq 9) ↓	ATE (Seq 10) ↓
Monodepth2	Self-Supervised	0.017	0.014
RelPose	GT Poses	**0.009**	**0.008**
Monodepth2 w/ RelPose (Ours)	Self-Supervised	1.091	0.779

Table 1: Overview of Results. Evaluation of Monodepth2 and RelPose on KITTI Odometry dataset. We achieve better average trajectory error (ATE) with supervised RelPose, and observe significantly worse performance when RelPose is trained in the Monodepth2 self-supervised framework.

We demonstrate the results of our proposed method in Tab 1. In order to establish baselines, we first train Monodepth2 on the KITTI Odometry dataset, and observe an average trajectory error of 0.017 on Sequence 9 and 0.014 on Sequence 10. Next, in order to examine whether RelPose might work on challenging scenes from KITTI, we train RelPose in a fully-supervised setting and achieve a strong performance of 0.009 and 0.008 on the two sequences respectively, which outperforms the Monodepth2 baseline. This is an interesting result which suggests that out-of-the-box RelPose is capable of generalizing to scenes with moving objects, despite it's epipolar inductive bias. A large part of the improvement, however, likely comes from the ground-truth pose supervision, whereas Monodepth2 is just simply using the self-supervised losses.



(a) Monodepth2 with PoseCNN



(b) Monodepth2 with RelPose

Figure 3: Monodepth2 Loss Curves. Training (orange) and validation (blue) losses for Monodepth2 with baseline PoseCNN and our addition of RelPose

Finally, our proposed method which introduces RelPose into Monodepth performs very poorly with an approximate 1000$\times$ worse trajectory error on both sequences. To understand this further, we first plot the loss in Fig 3, which show that the proposed network is not training (train loss is fairly constant). This leads us to investigate the reasons into this failure, and draw insights into the self-supervised monocular depth estimation framework. Results from our investigation are listed in Tab 2 and explained in further sections.

Method	Supervision	ATE (Seq 9) $\downarrow$	ATE (Seq 10) $\downarrow$
RelPose	GT Poses	**0.009**	**0.008**
Monodepth2	Self-supervised	0.017	0.014
Monodepth2 w/ RelPose (Final Layer training)	Supervised initialization, self-supervised final layer	0.022	0.021
Monodepth2 w/ RelPose	StreetLearn pretrained, Self-supervised	0.181	0.136
Monodepth2 w/ RelPose	Self-supervised + SuperGlue	0.029	0.023

Table 2: Our ablations to get a good ATE using the proposed network. Clearly, reprojection losses are not sufficient to train RelPose in a self-supervised setting. Using additional constraints such as SuperGlue loss alleviates the issue.

4.1 Verifying the correctness of our implementation

In order to verify that the proposed method was implemented correctly, we conduct experiments to simplify the number of parameters for the model to learn. We initialize RelPose from it's supervised pretrained weights on KITTI, and fine-tune only the final layer. Since it is trained on the same dataset, failure to obtain similar results as the second row in Tab 1 would suggest incorrect implementation. However, we observe very similar performance as seen in the third row of Tab 2, which suggests the issue lies elsewhere. Next, we use pretrained weights from StreetLearn instead, which is another outdoor dataset, and train the entire network end-to-end. We observe worse results compared to KITTI pretraining (which is expected), but still better than self-supervised approaches.

4.2 Pretrained initialization to bootstrap RelPose

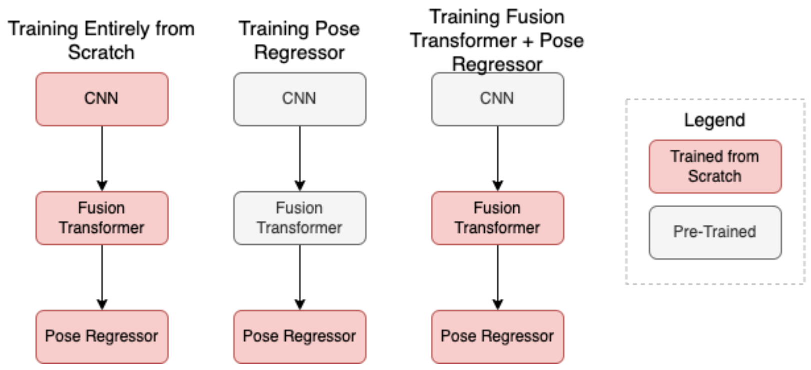


Figure 4: Bootstrap Experiments w/ RelPose. Different configurations of the RelPose network where we choose to initialize parts of the network from scratch or using pre-trained supervised weights.

In this experiment, we aim to assess whether adding a good initialization to RelPose helps convergence. We previously tried pretraining the entire network on KITTI and StreetLearn datasets. To investigate deeper into the root cause of where self-supervised loss isn't sufficient, we initialize some parts of the network from scratch and others with pre-trained weights from KITTI and then fine-tune the entire network as shown in Fig 4. Concretely, we aim to understand *which part* of RelPose is difficult to train under the self-supervised framework. To this end, we decompose RelPose into three main components: (a) CNN that extracts visual features, (b) fusion transformer that contains the ViT and essential matrix module to perform cross-attention using the extracted CNN features, and (c) the pose regressor to return the predicted pose.

Therefore, this experiment also aims to answer the question *whether it is even possible* to train RelPose in a self-supervised manner. In other words, if this experiment fails, this suggests that the self-supervision is not strong enough to train a vision transformer, in it's current form.

CNN	Fusion Transformer	Pose Regressor	ATE (Seq 9) $\downarrow$	ATE (Seq 10) $\downarrow$
Scratch	Scratch	Scratch	1.091	0.779
Pre-Trained	Pre-Trained	Scratch	**0.022**	**0.021**
Pre-Trained	Scratch	Scratch	0.556	0.397

Table 3: Bootstrap ablations. Different configurations where we initialize from scratch versus using pretrained weights. Our performance increases significantly when we use pretrained transformer weights compared to training from scratch.

Results from our bootstrap ablations are shown in Tab 3. We first trained the entire model from scratch, and observed the worst performance. We then observe a steep difference in performance when using the pretrained fusion transformer compared to training from scratch as shown in rows two and three. This implies that *training the transformer is the bottleneck* in the self-supervised framework. A question the reader may have is whether the fusion transformer can be trained from scratch while using pretrained initialization for the pose regressor and CNN feature extractor. This could unequivocally show that the vision transformer is the source of failure in the self-supervised setting. In theory this could be done, but our intuition suggests that since the pose regressor depends on the output of the transformer, this may lead to a distribution shift (especially during early stages of training) which may result in unstable training and potentially poor results.

We conducted experiments with different hyper-parameters, optimizers, and learning rate schedulers, but observe similar results. This is rather surprising, but suggests that current self-supervised losses for pose estimation and depth estimation networks are not sufficient for training vision transformers such as RelPose. We believe further investigation and potentially performing a deep neural architecture search may be beneficial as next steps to strengthen our hypothesis. While not the scope of this report, we demonstrate that future self-supervised learning research should aim to impose stronger geometric constraints to yield better performance.

4.3 Pseudo-Labels with SuperGlue

The natural next question is whether we can exploit some other signals for improved self-supervised training. In order to closely mimic the supervised RelPose setting (which is proven to work by us in earlier results), we introduce a pseudo-label generation mechanism for weak supervision. Concretely, given pairs of images, we compute the relative pose using matches from SuperGlue [5] and use these as additional supervision for RelPose. We replicate the RelPose training strategy within the self-supervised framework, by introducing a geodesic loss between the predicted and computed relative poses. However, if the pose cannot be estimated using an essential matrix decomposition (e.g: less than 5 matches, low inlier count, images explained by a homography), we do not apply the loss to those images.

The results are reported in the final row of Tab 2. This methods achieves a trajectory error of 0.029 and 0.023 respectively on sequences 9 and 10, which is comparable (but still worse) than Monodepth2's results. We note that tuning the loss weights could lead to improved performance, but we do not explore this and set the weight to 0.1. However, this investigation suggests that additional stronger constraints may be required to train RelPose well in a self-supervised manner.

Is this *really* self-supervised? While using SuperGlue for pseudo-labels is one approach for additional self-supervision, some may argue that this is not really in the spirit of self-supervised learning. They may claim SuperGlue is trained on wide variety of data, and using pseudo-labels generated from it assumes access to a method which was supervised. We do accept that SuperGlue supervision may provide ground truth supervision indirectly, due to it's strong performance on outdoor scenes. However, through our experiments, we hope to spark a discussion to find other constraints which could help training transformers in this self-supervised setting. The introduction of the SuperGlue-based loss shows that correspondences can help significantly improve the performance of our pose prediction.

5 Conclusion

In this work, we aim to investigate whether a transformer-based method such as RelPose can be trained in a self-supervised manner. Using the Monodepth2 framework, we supervise RelPose using depth and photometric reprojection losses. On observing poor results, we thoroughly investigate the root cause of the issue, with our experiments suggesting that reprojection losses are not sufficient to train RelPose. We show that providing additional supervision in the form of SuperGlue correspondences aids learning significantly. Our experiments provides key insights into training vision transformers in self-supervised pose predictions settings, and we hope to inspire future work on stronger geometric constraints as self-supervision.

6 Future Work

We find the results of our investigation very promising and would like to continue to explore this further. Transformer-based pose prediction models may require stronger supervision compared to CNN-based models due to their inductive bias. Even despite the epipolar constraints encoded within the transformer, we see that typical reprojection loss is insufficient for the model to learn. We would like to replicate this result on other datasets (besides KITTI), such as Matterport and StreetLearn to validate our hypothesis. Matterport and StreetLearn are supported by RelPose but not by Monodepth2 and therefore, require some engineering effort. Furthermore, we need to perform a full hyper-parameter search, and tweak our optimization strategy as we are introducing a transformer in our Monodepth2 model which evidently does not work well in its current form and likely requires significantly different training recipes. Lastly, we would like to investigate other self-supervised losses that synergize better with transformers.

References

[1] J. L. Schonberger and J.-M. Frahm, "Structure-from-motion revisited," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4104–4113, 2016.

[2] C. Rockwell, J. Johnson, and D. F. Fouhey, "The 8-point algorithm as an inductive bias for relative pose prediction by vits," in *3DV*, 2022.

[3] C. Godard, O. Mac Aodha, M. Firman, and G. J. Brostow, "Digging into self-supervised monocular depth estimation," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 3828–3838, 2019.

[4] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[5] P.-E. Sarlin, D. DeTone, T. Malisiewicz, and A. Rabinovich, "Superglue: Learning feature matching with graph neural networks," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 4938–4947, 2020.