Pattern-based pose estimation for Tactile Internet

Marijn Leo Louis Craenen[1]

Supervisor(s): Ranga Rao Venkatesha Prasad[1]**, Kees Kroep**[1]

[1]**EEMCS, Delft University of Technology, The Netherlands**

A Thesis Submitted to EEMCS Faculty Delft University of Technology,
In Partial Fulfilment of the Requirements
For the Bachelor of Computer Science and Engineering
June 25, 2023

Name of the student: Marijn Leo Louis Craenen
Final project course: CSE3000 Research Project
Thesis committee: Ranga Rao Venkatesha Prasad, Kees Kroep, Michael Weinmann

Abstract

The Tactile Internet (TI) is a new paradigm for remote interactions, enabling the transmission of touch and physical sensations. One of the major challenges in achieving seamless remote interactions is latency. To circumvent strict latency requirements, the paper briefly introduces the approach of a Model Mediated Teleoperation scheme utilizing a locally run physics engine to simulate the remote environment.

The focus of this paper is on solving the problem of tracking objects in TI workspaces, to be able to simulate them. We developed a pattern recognition-based pose estimation technique using OpenCV's Perspective-n-Point solver, which accurately estimates the pose of objects in real time. Further contributions include the implementation of a virtual test bed in the Unity game engine. The solver and test bed were integrated with a Python Flask server. This approach proved to be effective in providing accurate position and rotation estimation.

1 Introduction

The concept of the Tactile Internet (TI) is emerging as a new remote interaction paradigm, aiming to revolutionize remote interactions by enabling the transmission of touch and physical sensations. The tactile internet holds the promise of bridging geographical distances and creating a sense of physical presence between users or between users and machines, by allowing seamless teleoperation over long distances.

The objective of the Tactile Internet is to provide real-time, high-fidelity haptic feedback over the internet, surpassing the boundaries of traditional information exchange.

One fundamental challenge in achieving seamless remote interactions lies in latency, the unavoidable delay in transmitting data over long distances [1]. As the speed of light sets the lower bound for latency, there are inherent limitations to how far teleoperation can effectively be carried out. However, innovative approaches have emerged to overcome this hurdle. Rather than attempting to reduce latency beyond the speed of light, a viable solution lies in observing the remote environment and *simulating* it with a locally run physics engine, allowing users to interact without network latency on the simulation, rather than directly with the remote environment. This novel *Model Mediated Teleoperation* (MMT) scheme is illustrated in Fig. 1.

For accurate and realistic feeling control, it is essential to have precise and up-to-date observations of the workspace in real time. Objects in the workspace need to be tracked as they are encountered or interacted with. Conventional RGB cameras provide a common, affordable, and accessible imaging solution, but the simple 2D images they create face challenges when it comes to object tracking without depth sensors or advanced machine-learning techniques.

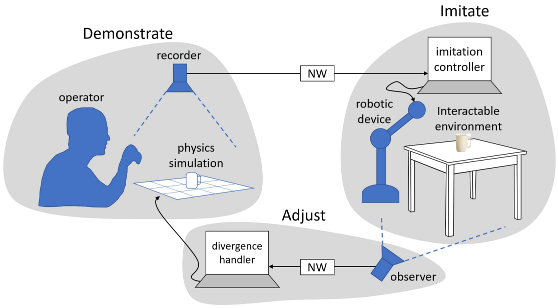


Figure 1: An illustration of Tactile Internet. The operator in the master domain sends instructions to the controlled domain through a network by interacting with a simulation. The controlled domain's workspace is observed and feedback from the taken actions is fed back to the master domain's simulation. ***Illustration by Kees Kroep.***

To address this obstacle, this paper investigates pattern recognition-based pose estimation techniques, where a pattern of known dimensions, such as a checkerboard pattern on a sticker, is used as a reference for tracking objects. This approach enables accurate tracking and position estimation, enhancing the overall fidelity of remote interactions within the Tactile Internet framework.

This paper is part of ongoing research on MMT using a physics engine for Tactile Internet, at the TU Delft.

Contributions The objective of this paper is to present the following contributions, which aim to solve object tracking in TI workspaces:

1. **Virtual test bed:** We implemented a virtual test setup in the Unity game engine that allows for rapid prototyping and testing of RGB camera tracking in a 1:1 real-world scale.
2. **Pose estimation algorithm:** The pose estimation algorithm implementing OpenCV's Perspective-n-Point (PnP) solver takes in an image and parameters and returns an accurate pose estimation, providing both position and rotation estimations.
3. **Integration framework:** The virtual test bed and the pose estimation are integrated by a Flask server API called from the Unity test bed, allowing the posing process to be fully executed from the Unity editor without even needing to enter Play mode.

The paper is organized as follows: Section 2 discusses relevant works in the fields of TI, MMT, and pose estimation methods. Section 3 explains the methodology and considerations followed during research. Section 4 then concretely specifies the exact experimental setup. This is followed by the results of the experiments in section 5. Section 6 discusses further context and acknowledgments related to the results, and section 7 gives recommendations for future research. The ethical implications of this work are mentioned in section 8 before the paper concludes with section 9.

2 Related works

2.1 Tactile Internet

Tactile Internet was introduced by Fettweis in 2014 [1]. While lots of research has been done on the subject of latency reduction, no publications have yet taken the approach using a local physics engine to circumvent stringent latency requirements [2] [3] [4]. State-of-the-art research is relying on the latency and reliability improvements promised by 6G networks. Some of these implementations use 6G ultra-low latency and high-reliability networks in combination with a feedback control system, to relax the latency requirements [5] [6] [7]. Using a control system is however a distinctly different approach than using a physics simulation, which is the basis for this paper.

2.2 Camera calibration and pose estimation from pattern

Camera calibration from a pattern on a 2D plane is a technique first introduced by Zhang in 2000, which has since been firmly established as a fundamental tool for camera calibration as well as pose estimation [8]. While many patterns can be used to solve the Perspective-n-Point problem, a checkerboard pattern is a simple pattern that has been incorporated into the OpenCV standard library [9]. It is for this reason that the checkerboard pattern was used in the prototype presented in this paper.

2.3 Alternative pose estimation methods

Alternative methods to pattern recognition for 3D pose estimation from 2D images exist. Many of these methods are based on the SIFT feature extraction algorithm developed by Lowe in 2004 [10]. SIFT represents objects as a sparse set of invariant features computed from training images, which can then be detected in new images.

Since depth (RGB-D) cameras became more prevalent, many algorithms have been developed that do pose estimation using RGB-D cameras in combination with machine learning. These include using a regression forest on a known scene to infer camera pose [11], and templating, where a small image patch is compared with subregions of a larger image to find areas that closely match [12] [13].

Another popular approach is CAD-based models [14]. This method relies on prior knowledge of the objects to detect in the form of a 3D CAD model. An approach that does not require prior knowledge is pose estimation from video using Structure-from-Motion imaging techniques [15].

The aforementioned approaches were considered for the problem of object pose estimation in a TI workspace, but were ultimately deemed unsuitable for this bachelor thesis. With the strict time restraints, a simple approach was necessary without machine learning. The need for a simple and cheap prototype for object detection and posing led to the exploration of the possibilities offered by a conventional RGB camera.

3 Methodology

This section explains the design choices which were made to solve the object tracking problem using RGB cameras, starting with an exploration of point cloud generation, transitioning to pattern recognition using a PnP solver, and culminating in the automation of the pose estimation process. The Unity environment served as the foundation for the experimental setup, enabling programmable camera rig construction and facilitating the evaluation and visualization of results. The following subsections discuss the approach in detail, step by step.

3.1 Virtual test bed

Unity game engine was chosen as the virtual environment for this study over the use of real cameras, due to its ability to provide a highly configurable and consistent testing platform. By leveraging Unity, every aspect of the experimental setup could be precisely configured, allowing for fine-tuning and adjustments of variables as needed. The virtual environment ensured that all parameters, such as camera properties and lighting conditions, remained consistent throughout the experiments, eliminating potential confounding factors. This way the study ensured a controlled and reproducible environment, facilitating accurate analysis and evaluation of the pose estimation technique within the tactile internet context.

3.2 Point cloud generation

Initially, an attempt was made to do pose estimation through the generation of point clouds from an array of 2D images. For this purpose, a programmable camera rig was developed in Unity, providing flexibility and control over the cameras' behaviors. An array of cameras was set up in a circle pointing at the workspace center point. Through photogrammetry techniques, their picture output was then used to create a 3D point cloud of an object in the workspace. It was found however that this approach proved to be far too slow and resource-intensive for use within the TI context. The programmable camera approach did allow for more efficient and cost-effective experimentation, which proved beneficial for subsequent advancements in pose estimation techniques.

3.3 Perspective-n-Point pattern recognition

The focus shifted towards pattern recognition, by utilizing the Perspective-n-Point (PnP) algorithm.

PnP is a method in computer vision that estimates the pose (position and orientation) of an object based on its 2D projection in an image and the known 3D coordinates of corresponding points on the object. This algorithm relies on the principle of finding the correspondences between the 2D image points and their corresponding 3D points in a calibrated coordinate system.

The process of pattern recognition using PnP involves several steps. Initially, a calibration step is performed to determine the intrinsic and extrinsic parameters of the camera. The intrinsic parameters include the focal length, principal point, and distortion coefficients, while the extrinsic parameters define the position and orientation of the camera with respect to the world coordinate system. To do this, a series of pictures of a checkerboard was taken in various positions and angles in the virtual workspace, which resulted in a mean re-projection error of less than $0.08\,\mathrm{pixels}$. 4 examples of re-projection of this calibration onto the checkerboard can be seen in Fig. 2.

Once the camera is calibrated, the next step is to detect and extract the pattern of interest, such as a checkerboard or ArUCo markers, from the captured image. The detected pattern is then matched with the known 3D coordinates of the corresponding points on the pattern. This correspondence information serves as the input to the PnP algorithm.

The PnP algorithm calculates the pose of the pattern by finding the transformation matrix that aligns the 3D coordinates of the pattern points with their corresponding 2D image points. The transformation matrix represents the translation and rotation required to align the pattern with the image.

The objective of this paper was to make the posing of checkerboards work reliably, serving as a comprehensive prototype before further research progresses to the utilization of more practical ArUCo markers on objects.

3.4 Automated testing

To streamline the process and enhance efficiency, the entire methodology was automated using a system of preset poses in the virtual testbed. The poses were automatically iterated and photographed, and the photos were sent to a Python server that executed the actual pose estimation. This seamless integration allowed for automated data processing, pose estimation and result retrieval in a reproducible way. By eliminating manual intervention, this automation aimed to reduce human error and increase the scalability of the pose estimation system. Through this method, thousands of poses in one preset were able to be estimated in a matter of minutes.

Parameter	Value
Camera resolution	1920×1080
Checkerboard grid	10×7
Square size	$5\,\mathrm{mm}$
PC Processor	Intel i7-8750H @2.2GHz
PC RAM	$24\,\mathrm{GB}$

Table 1: The exact parameters used in the experimental setup.

3.5 Ground truth

The automated presets allow for easy retrieval of ground truth pose information. The precise placement of the checkerboard with respect to the virtual camera was known, ensuring an exact reference for evaluating the accuracy of the pose estimation. This is written to a CSV file. In the Python backend, this can then be compared with the pose estimates to obtain an error value.

4 Experimental setup

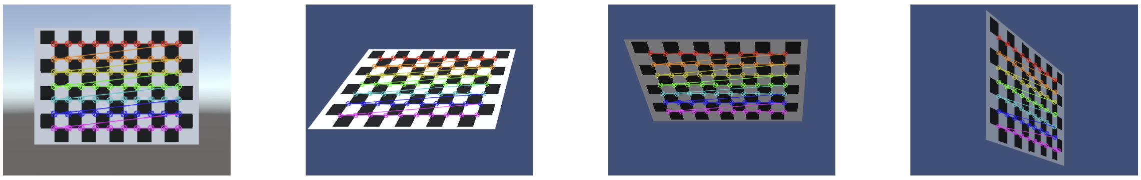
4.1 Testing parameters

A default virtual RGB camera within the Unity environment was utilized. The checkerboard pattern was placed on an A4-sized plane, and then scaled down by a factor 5 such that the squares were $5\,\mathrm{mm}$. All the parameters used can be referenced in Tab. 1. Lighting was done by Unity's Sample Scene's default *Directional Light* with an intensity of 1. For the camera background both the default skybox and a flat blue shade were used, as seen in Fig. 2. In Fig. 3 you can see the final test bed setup.

4.2 Pose estimation

Multiple presets were configured programmatically to study position and rotation estimation performance:

1. **Repeated samples, same pose.** This preset verified that the PnP solver is deterministic for the same picture input. The preset had 30 samples of the checkerboard facing the camera directly at $20\,\mathrm{cm}$ distance.
2. **Sweeping distance 0.15 m - 0.75 m.** This preset was built to investigate the pose estimation accuracy as the distance to the camera increases. Since pose estimation is deterministic, capturing multiple samples at the same distance alone may not provide sufficient variation to observe a meaningful variance in the results. Instead, multiple samples were obtained by strategically varying the X/Y position of the checkerboard on the screen while maintaining the same distance. This approach allowed for the collection of a diverse dataset, better capturing the system's behavior in a broader range of scenarios. It provided insights into the robustness and consistency of the pose estimation algorithm.



(a) (b) (c) (d)

Figure 2: Four examples of the re-projection of checkerboard corners detected by the camera calibration process onto the original checkerboard image. This is strictly for visual verification that the calibration is working properly and is not necessary for the calibration process itself. The input pictures to the camera calibration process used both Unity's default *"skybox"* and a flat blue shade as the camera background, both are pictured here.

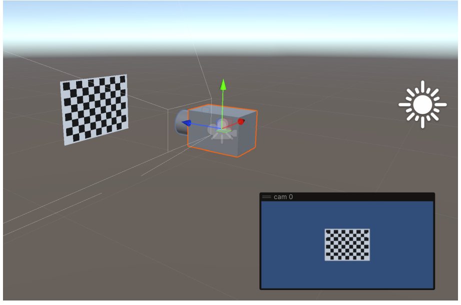


Figure 3: The virtual test bed in Unity, showing the camera pointed at the checkerboard, the directional light behind it, and a preview of the camera's view in the bottom right corner.

6 X-coordinates and 6 Y-coordinates were used, making 36 unique screen positions for each of the 25 Z values studied, giving a total of 900 samples.

3. **Sweeping roll, pitch, yaw.** This preset has the checkerboard rotated between -60 and 60 degrees on the pitch axis, then similarly rotated on the yaw axis, and finally rotated -180 to 180 degrees on the roll axis. Rotation was swept in steps of 5 degrees. The regions -90 to -60 and 60 to 90 degrees were skipped for the pitch and yaw sweeps because no checkerboards could be detected by the PnP solver for those angles. The total sample count for this set was 5040, with 1008 for both pitch and roll and 3024 for yaw, all at a distance of $0.3\,\mathrm{m}$.

4.3 Server integration

The integration between the Python posing algorithm and the Unity test bed is managed by a Python Flask server running locally. The Unity test bed sends a POST request to the server's API with the picture data and parameters, and the server pipes the data through the PnP solver. Its output is stored on the local hard drive and optionally sent back to the Unity test bed. The test bed can then visualize the pose by projecting an axis system onto the checkerboard in the scene, as illustrated in Fig. 4. The axis system is projected onto the top left corner of the checkerboard as that is considered its origin.

5 Results

The resulting data of pose estimation of the checkerboard in 6 degrees of freedom are presented here.

Due to the deterministic nature of the PnP solver algorithm as proven by the *"Repeated samples, same pose"* test configuration, sweeping a variable such as distance or angle on a checkerboard pattern positioned in the center of the frame did not yield diverse or informative results. In order to explore a wider range of scenarios and capture more nuanced information, it became necessary to introduce more complex patterns or incorporate additional factors into the pose estimation process. Therefore for each data point of the swept variable, the checkerboard was photographed in multiple X/Y positions in the camera frame. This gives insight into the variance of pose estimation error for objects that might appear in any part of the screen.

5.1 Position

In Fig. 5 the variance of estimation error is plotted, as the checkerboard moves away from the camera. Each axis is individually plotted. The observed variance is less than $0.1\,\mathrm{mm}$ off for the X and Y axes. The Z-axis mostly follows the near perfect accuracy of the other axes, with a spike in variance between $0.325\,\mathrm{m}$ and $0.55\,\mathrm{m}$ that does not exceed $1\,\mathrm{mm}$. At distances beyond $0.7\,\mathrm{m}$ the variance spikes drastically as samples start to fail to converge. Beyond $0.75\,\mathrm{m}$ the PnP algorithm fails to detect any checkerboards.

The achieved submillimeter accuracy is noteworthy, even surpassing the stringent requirements for precise surgical operations as established in related literature [16]. It highlights the promising potential for achieving consistently high positional accuracy in the pose estimation prototype. However, to ensure a comprehensive understanding of the system's performance, further investigation is required to determine the underlying cause of the spike along the Z-axis.

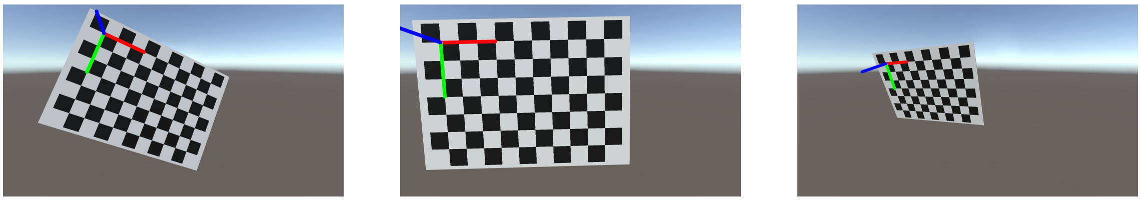


Figure 4: Results of pose estimation re-projected as a Cartesian coordinate system on the checkerboard in the Unity test bed for visualization purposes. The X-axis is shown in red, Y-axis in green and Z-axis in blue.

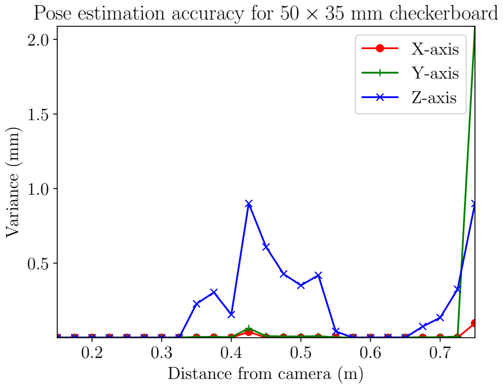


Figure 5: The pose estimation algorithm's accuracy is plotted for various distances (points along the Z-axis) of the checkerboard from the camera. Multiple samples were taken per Z-axis point by having the checkerboard at multiple distinct X/Y positions in the camera frame. The variance of these samples per Z-axis point is plotted.

One plausible explanation is that distortions in the checkerboard squares or dimensional variations caused by aliasing, resulting from the camera's resolution, could contribute to this phenomenon. Aliasing artifacts change with distance to the camera, so that might explain why the variance decreases again after 0.55 m. Addressing this aspect will contribute to improving the overall accuracy and reliability of the pose estimation system.

5.2 Rotation

For the following results, 0 degrees rotation is the checkerboard rotation as seen in Figs. 2 (a) and 3.

The variance of the angular error is plotted in Fig. 6 and Fig. 7 for the roll, pitch and yaw angles. The horizontal axis shows the angle with respect to the camera, and the vertical axis the error variance of the samples on various x/y positions with that angle. The error is very low, arguably negligible between -45 and 45 degrees for the pitch and roll axes. As the angle to the camera increases beyond that the error spikes, particularly on the roll axis. Beyond ± 50 degrees many samples also fail to converge. This could be correlated with the higher variance observed there.

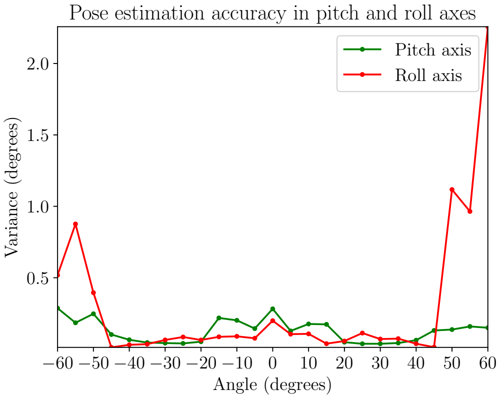


Figure 6: Pose estimation accuracy for the pitch and roll axes, corresponding to the Y and X axes in Fig. 4 respectively. The rotation was done between $-90°$ and $90°$ but since beyond $60°$ no checkerboards could be detected in the samples they are excluded from the plot.

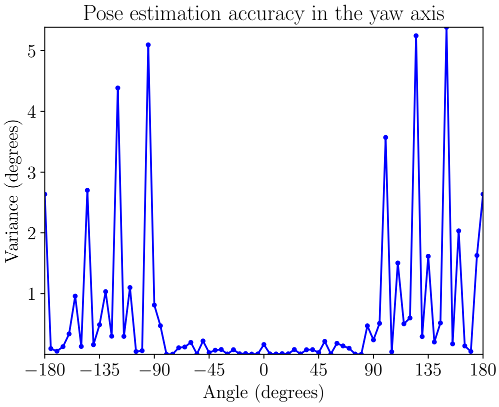


Figure 7: Pose estimation accuracy for the yaw axis, corresponding to the Z axis in Fig. 4. The rotation was done between $-180°$ and $180°$. All samples converged to a detected pose.

The pose estimation on roll is almost perfectly accurate between -90 and 90 degrees, and then starts to experience pretty significant variance spikes. It is not clear why this might happen, as the checkerboard is fully and distinctly in view with the exact same lighting conditions as the samples where accuracy is high. It is worth noting that the significant spikes in variance occur only when the checkerboard is rotated beyond 90 degrees, i.e. when it is more upside down than upright. The reasons behind this behavior, despite the checkerboard being visible and the lighting conditions remaining consistent, are not clear yet and require further investigation.

5.3 Frame rate

The average frame rate achieved for pose estimation in the experiments was 40 Hz. This was achieved on the system specifications listed in Tab. 1. This frame rate surpasses the standard frame rate of the Kinect camera at 30 Hz typically utilized in RGB-D implementations, as well as the typical default video frame rates. The obtained frame rate of 40 Hz is nowhere close to the 1 kHz frequency used in the MMT physics simulation, but it might be used complementarily.

5.4 Summary

The results obtained from the experimental evaluations highlight the strengths and limitations of the prototype developed for TI teleoperation. The high positional accuracy and consistent performance within a realistic distance range demonstrate there is a lot of potential in the prototype for precise teleoperation tasks. However, challenges in accurately estimating poses at extreme rotational angles need to be addressed or circumvented, for example through the use of multiple checkerboards on different faces of an object, to enhance the prototype's overall performance and broaden its application in real-world TI scenarios. Finally, it is worth mentioning that the current implementation of the system achieves an average framerate of 40 Hz for pose estimation, which provides a satisfactory real-time performance for various applications, surpassing the default framerates of most video sources and even the Kinect camera's 30 Hz.

6 Discussion

This research had a strict time restraint of about 9 weeks. This put considerable scope restraints on the research. To exacerbate the problem, the initial research topic was considering mesh completion from partial point clouds generated by a depth camera. This topic, after some research, was not interesting to me. It was only in week 2 that I switched to the topic of tracking objects using conventional cameras. Once that was established, I spent too much time digesting papers on object detection and pose estimation without having a general understanding of the topics or a clear idea of what to look for. I went through a steep learning curve of how research is done, and this cost me valuable time.

Unfortunately, for two weeks during the research project, I've been dealing with a nasty fever that took a toll on me. My productivity took a nosedive because I just couldn't work through the symptoms and mental fog. It was a struggle to even get started on tasks that usually come easily to me. It was frustrating to see my to-do list grow while my energy levels plummeted. I tried my best to push through, but eventually, I had to admit that I needed to adjust the scope of my project. The things I would have liked to have gotten done are mentioned in the next section.

7 Future work

Despite the time constraints imposed on this research, several stretch goals were envisioned but could not be pursued within the given timeframe. One such goal was the implementation of ArUCo marker tracking, which offers practical advantages in the context of Tactile Internet (TI) teleoperation. ArUCo markers enable the tracking of multiple distinct stickers on an object, providing finer-grained information about its pose and enhancing the accuracy of teleoperation. An object might have stickers on all sides so it can be tracked from any angle. Unfortunately, due to the limited duration of the project, it was not feasible to explore this avenue.

An alternative avenue to investigate is the utilization of multiple cameras to enhance the reliability of pose estimation. A multi-perspective approach can facilitate a more robust and accurate pose estimation, as it combines information from different vantage points, effectively mitigating occlusions and ambiguity. Having access to multiple angles might greatly enhance the tracking process.

Exploring the integration of machine learning techniques for object detection and pose estimation could also significantly enhance the capabilities of TI teleoperation. Training deep learning models on large-scale datasets to recognize and track objects in real-time could provide more accurate and robust pose estimation, thereby improving the overall performance of TI systems.

It would also have been very interesting to expand on the current research on checkerboards. For instance investigating camera resolutions; how high it needs to be for accurate tracking. Another interesting idea is to compare the results from this research with depth (RGB-D) camera and inertial measurement unit tracking implementations presented by other members of the TU Delft TI research group.

Although certain stretch goals could not be realized within the given time constraints, the research outcomes have still shed light on the challenges and possibilities of object tracking in TI teleoperation.

8 Responsible Research

This study did not involve the collection, use, or analysis of any personal or sensitive data related to individuals. Therefore, privacy violations and associated ethical concerns typically associated with data usage and handling were not applicable to this research.

It is important to note that while ChatGPT was utilized to assist in the writing process of this paper, it played a supportive role in generating draft content. This aided in accelerating the paper writing process. The final composition and structure of the paper was determined by the author, who carefully reviewed, augmented and rewrote the generated text to ensure its accuracy and adherence to responsible research standards.

We recognize the need for transparency and integrity in scientific research, and we have strived to uphold these principles throughout the course of this study. We have made every effort to cite and reference relevant sources appropriately, giving credit to the contributions of other researchers in the field.

9 Conclusion

The Tactile Internet (TI) is a new paradigm for remote interactions by enabling the transmission of touch and physical sensations. The ultimate goal is to provide real-time, high-fidelity haptic feedback over the internet. One of the major challenges in achieving this seamless remote interaction is latency, an unavoidable delay in transmitting data over long distances with the speed of light setting an absolute lower bound.

To overcome this hurdle, the paper introduces the approach of a Model Mediated Teleoperation scheme utilizing a locally run physics engine to simulate the remote environment. This allows users to interact with the simulation without network latency, improving the overall fidelity of remote interactions within the TI framework.

In this paper, we focused on solving the problem of tracking objects in TI workspaces. We investigated pattern recognition-based pose estimation techniques using a checkerboard pattern as a reference for tracking objects. This approach proved to be effective in providing accurate tracking and position estimation.

The contributions of this paper include the implementation of a virtual test bed in the Unity game engine for rapid prototyping and testing of RGB camera tracking. We also developed a pose estimation algorithm using OpenCV's Perspective-n-Point solver, which accurately estimates the pose of objects in real time. Additionally, we integrated the virtual test bed and pose estimation algorithm through a Flask server API, allowing for seamless execution of the posing process from the Unity editor.

Our methodology involved automated testing and data processing, which improved efficiency and reduced human error. The virtual environment provided a controlled and reproducible setting for evaluating the pose estimation technique within the TI context.

While our experiments focused on the posing of checkerboards, serving as a comprehensive prototype, we acknowledge the potential of using more practical ArUCo markers on objects in future research.

In summary, this paper contributes to the ongoing research on physics-based MMT for the Tactile Internet by addressing the object tracking challenge in TI workspaces. Our findings demonstrate the feasibility and effectiveness of pattern recognition-based pose estimation techniques using RGB cameras. This research opens up possibilities for enhancing remote interactions by providing more accurate and realistic haptic feedback.

References

[1] G. P. Fettweis, "The tactile internet: Applications and challenges," *IEEE Vehicular Technology Magazine*, vol. 9, no. 1, pp. 64–70, 2014.

[2] M. Simsek, A. Aijaz, M. Dohler, J. Sachs, and G. Fettweis, "5g-enabled tactile internet," *IEEE Journal on Selected Areas in Communications*, vol. 34, no. 3, pp. 460–473, 2016.

[3] J. Sachs, L. A. A. Andersson, J. Araújo, C. Curescu, J. Lundsjö, G. Rune, E. Steinbach, and G. Wikström, "Adaptive 5g low-latency communication for tactile internet services," *Proceedings of the IEEE*, vol. 107, no. 2, pp. 325–349, 2019.

[4] V. Gokhale, M. Eid, K. Kroep, V. Prasad, and V. Rao, "Toward enabling high-five over wifi: A tactile internet paradigm," *IEEE Communications Magazine*, vol. 59, pp. 90–96, 12 2021.

[5] G. P. Fettweis and H. Boche, "6g: The personal tactile internet—and open questions for information theory," *IEEE BITS the Information Theory Magazine*, vol. 1, no. 1, pp. 71–82, 2021.

[6] Z. Hou, C. She, Y. Li, D. Niyato, M. Dohler, and B. Vucetic, "Intelligent communications for tactile internet in 6g: Requirements, technologies, and challenges," *IEEE Communications Magazine*, vol. 59, pp. 82–88, 12 2021.

[7] A. A. Ateya, A. Muthanna, A. Vybornova, I. Gudkova, Y. Gaidamaka, A. Abuarqoub, A. D. Algarni, and A. Koucheryavy, "Model mediation to overcome light limitations—toward a secure tactile internet system," *Journal of Sensor and Actuator Networks*, vol. 8, p. 6, Jan 2019.

[8] Z. Zhang, "A flexible new technique for camera calibration," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 11, pp. 1330–1334, 2000.

[9] OpenCV, "Perspective-n-point (pnp) pose computation." `https://docs.opencv.org/4.x/d5/d1f/calib3d_solvePnP.html`.

[10] D. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, pp. 91–110, 2004.

[11] J. Shotton, B. Glocker, C. Zach, S. Izadi, A. Criminisi, and A. Fitzgibbon, "Scene coordinate regression forests for camera relocalization in rgb-d images," in *2013 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2930–2937, 2013.

[12] S. Hinterstoisser, V. Lepetit, S. Ilic, S. Holzer, G. Bradski, K. Konolige, and N. Navab, "Model based training, detection and pose estimation of texture-less 3d objects in heavily cluttered scenes," vol. 7724, 10 2012.

[13] S. Hinterstoisser, C. Cagniart, S. Ilic, P. Sturm, N. Navab, P. Fua, and V. Lepetit, "Gradient response maps for real-time detection of textureless objects," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 5, pp. 876–888, 2012.

[14] M. Ulrich, C. Wiedemann, and C. Steger, "Cad-based recognition of 3d objects in monocular images," in *2009 IEEE International Conference on Robotics and Automation*, pp. 1191–1198, 2009.

[15] J. Sun, Z. Wang, S. Zhang, X. He, H. Zhao, G. Zhang, and X. Zhou, "Onepose: One-shot object pose estimation without cad models," 2022.

[16] T. Haidegger, L. Kovács, B. Benyó, and Z. Benyó, "Spatial accuracy of surgical robots," pp. 133–138, 05 2009.