Vision-Based Camera Motion Tracking

Design Document

Team Number: sddec25-15

Client: Eric Wittrock

Advisers: Prof. Ashraf Gaffar

Team Members: Will Ernatt, Andrew Gooding, Eric Wittrock, Isaac Kenyon

Team Email: sddec25-15@iastate.edu

Team Website: https://sddec25-15.sd.ece.iastate.edu/

Revised: 5/4/2025

Executive Summary

The goal of this project is to try and make camera tracking in visual effects creation less tedious. Camera tracking is the process of estimating the positions of the camera in 3D space over time provided its footage. This is necessary when compositing 3D elements onto 2D footage so the motion of the rendered objects match that of the real scene. Simplifying this process will save time and allow solo artists and small teams to create 3D effects at a lower cost.

The design requirements that were chosen involve constraints on simplicity, efficiency, and correctness. The algorithm will automate as much of the process as possible, and the UI should be minimal so that the learning curve is shallow as intended. The algorithm also should be performant so that the time it takes to complete is on par with competitors' solutions. Most importantly, the output should be correct so that there is no visible difference in motion between the real background and the 3D virtual objects.

Overall, the design is composed of three main parts. The first is a point tracking algorithm, which detects high contrast points in a video to observe the sparse motion of pixels. In this first attempt, a custom-made neural-network-based algorithm was used, but the performance was incredibly slow. Therefore it was decided to switch to OpenCV's built-in optical flow tool until a better solution could be found. The ORB-SLAM algorithm is a promising candidate, but if the final result is satisfactory, the current solution could be kept permanently. The next main component is the camera solving algorithm. Given the list of points and their translations over time, it gives a viable motion path that the camera could have taken. This algorithm was written from scratch in Python. The final component is the UI, which interacts with the aforementioned algorithms. It was written in Python as a Blender plugin.

Thus far, a working prototype currently calculates a camera track for a trivial scene with minimal points. Usability remains straightforward in the absence of edge cases such as moving objects. In these simple scenarios, results are satisfactory, with minimal error between the motion of 3D renders and the background. However, performance remains below expectations. Upcoming efforts will focus on improving processing speed. Integration of the ORB-SLAM algorithm with the optimization algorithm is planned to provide the ADAM optimizer with an improved initial guess, which is expected to enhance performance significantly. Machine learning has also been considered to generate more informed next-guesses within the optimization algorithm, aiming for similar improvements. The final development stage will involve adapting the solution to handle more complex scenes, including those with reflective surfaces and moving elements.

Learning Summary

Development Standards & Practices Used

Engineering Standards Applicable for this project

- IEEE 1680.1-2009 - IEEE Standard for Environmental Assessment of Personal Computer Products, Including Notebook Personal Computers, Desktop Personal Computers, and Personal Computer Displays
    - This standard defines environmental performance criteria for personal products, including desktop, laptop, and computer displays.
    - This standard is applicable to this project because it is intended for the project to be used on personal computers, desktops, and other small computers without having any issues with how it will affect computer hardware.
- IEEE Standard for Secure Computing Based on Trusted Execution Environment
    - This standard describes how computing systems and the developers of executables should act together. It describes standards and practices that the executable developers should follow to prevent security problems.
    - The Blender plugin ecosystem necessitates transparency for all 3rd party software. It is important that augmented software will not cause harm. For this reason, the execution environment standard is an extension of Blender's. A standalone executable is not an example of a viable final product because of the level of trust this would require.
- ISO/IEC/IEEE International Standard for Systems and Software Engineering - Recommended Practice for Architectural Description of Software-Intensive Systems
    - This standard describes how a detailed design document and other files should explain a complex software system. This is important for uniformity because software systems can have hundreds to thousands of components that are impossible for one person to understand.
    - The project will implement multiple software systems and libraries that will then be implemented together creating an entirely new process. Since the architecture of this project involves many moving parts it is important to follow practices so that the system is well documented. Smaller sections of the system could start affecting other parts of the system, resulting in problems being hard to locate.

Summary of Requirements

Functional requirements

- Solve motion of physical camera when given footage and assign that motion to a virtual Blender camera with little to none user intervention.
- The output of this should not have any jitter between 3D and real scenes.

Resource requirements

- Users should be able to run software on local hardware with computational and memory requirements similar to a standard pc.
- The software should easily work as a Blender plugin.

Aesthetic requirements

- Minimalistic visual appealing design will be easy for users to learn to use.

User experiential requirements

- Must have a simple UI that is easy for users to navigate.
- Users shouldn't have any trouble getting past the learning curve.

Applicable Courses from Iowa State University Curriculum

SE/CPRE 1850, SE/CPRE 1860, COM S 3190, COM S 3090, COM S 4720, SE 4210, SE 4160, SE 4170

New Skills/Knowledge acquired that was not taught in courses

Working on this project has given the team an opportunity to learn new skills that would not have been acquired otherwise through course work at Iowa State. This project has led the team to learn how to use softwares including, but not limited to Blender's API and OpenCV. Through the development journey, all members learn about camera tracking in terms of usage and technical intricacies.

Table of Contents

List of figures/tables

Figures

Tables

1. Introduction

1.1. Problem Statement

With Visual effects becoming an increasingly popular mode of storytelling, product showcasing, and artistic expression, camera tracking has become a more commonly faced hurdle. At the very root of any visual effects project is the process of camera tracking. Visual effects (VFX) is the process of integrating computer generated visuals into real-life camera footage. Oftentimes, these computer generated images come from 3D models. To composite these two media sources, the 3D model has to move with the real environment in the camera's footage, but this is only possible if the motion of the camera is known. Fortunately, the camera's motion can be reconstructed with computer vision algorithms that look at the footage. While the algorithms under the hood do much of the heavy lifting for an artist, the process of camera tracking is still monotonous. There is a learning curve to get reliable results and even for experienced artists, the process can be time consuming. The goal is to provide a minimal-effort solution for solo artists that is fairly robust and accurate as well. VFX is time consuming and therefore a significant business expense. The ending product will significantly cut down on the time required for the camera tracking step by completely automating the process of finding high-contrast points, culling high error tracks, and more.

1.2. Intended Users

The users of this product range from professional studios to hobbyists video creators. Blender users will also find that the product integrates seamlessly with their systems that they already use. It will help create a quick integration into current projects.

The benefits of using the product will be ease of use, time saving, and ability to integrate into past and current projects. Most VFX artists find that automating tasks helps speed up the headaches of manually editing a video. This project will result in the ability to quickly add objects into one's own recorded videos. It should be as easy as recording a video, uploading into the Blender plugin and clicking calculate, with a resulting video that is just as good if not better then placing an object in a scene manually.

1.3. Personas

A professional user needs a way to modify the resulting camera track because they must ensure it matches the standard of the project they are part of, and their own vision. They care about the ability to quickly integrate, change parameters quickly, and speed of the processing. The professional user will benefit by the quick integration and low processing time. The professional user will save money by not having to use powerful hardware for the computation, and not having to spend much time waiting for rendering.

A prototyper needs a way to quickly and easily put an 3D object into a video and change certain parameters because the result is meant to show minimum viable product, rough testing, and the

ability to make revisions fast. This prototyper might not initially care about perfect implementation when going through revisions as every parameter change means more waiting for the scene to render. This user will benefit by all the saved time when testing different ideas.

An independent user needs a way to easily use and understand the program because this will make it easier for them to use their creative minds in their projects. Oftentimes the independent user is not a professional and does not care about the implementation of the project. The independent user wants to be able to import video, select an object, and click run. This user benefits by not having to read lots of documentation and change around parameters in order to get a good video output.

2. Requirements, Constraints, And Standards

2.1. Requirements & Constraints

Functional requirements

- Solve the motion of a physical camera given its footage and assign that motion to the virtual camera in Blender with minimal user intervention.
- The motion output should be believable, that is to say, there should be no jitter between the 3D scene and the real scene. The metric of success for this capability is pixel error, which can be calculated by the computer (constraint)

Resource requirements

- The software should be able to run on the user's hardware locally. (constraint)
- The computational and memory requirements are those of the standard pc.
- The software should seamlessly be able to work with Blender as a plugin.

Aesthetic requirements

- The minimalist design will compliment the end goal of ease-of-use.
- The design will be visually appealing to users.
- The design will be clear to users and have a shallow learning curve.

User experiential requirements

- Must be easily accessible, with simple UI design that any user can navigate through without frustration.
- Users should be able to get past the learning curve quickly.

2.2. Engineering Standards

The importance of engineering standards is providing uniformity and compatibility across all industries and organizations. The standards can also help provide important safety and efficiency features. The standards attempt to look at how products will affect the world and mitigate problems that may arise. For engineers, the standards are used as a blueprint which helps provide reliable, safe, and cost effective results.

Aspects of all three reviewed standards can be incorporated into the project. For instance, the IEEE Standard for Environmental Assessment of Personal Computer Products, Including Notebook Personal Computers, Desktop Personal Computers, and Personal Computer Displays, may primarily apply to hardware, but remains relevant since the generated output video will ultimately be viewed on a display.

Planned modifications include ensuring safe execution within the Blender environment. Blender standards will be followed in conjunction with IEEE standards for secure computing. The IEEE International Standard for Systems and Software Engineering – Recommended Practice for Architectural Description of Software-Intensive Systems will be applied in the detailed design document to ensure all diagrams maintain universality.

Engineering Standards Applicable for the project

- IEEE 1680.1-2009 - IEEE Standard for Environmental Assessment of Personal Computer Products, Including Notebook Personal Computers, Desktop Personal Computers, and Personal Computer Displays
    - This standard defines environmental performance criteria for personal products, including desktop, laptop, and computer displays.
    - This standard is applicable to the project because it is intended for the project to be used on personal computers, desktops, and other small computers without having any issues with how it will affect computer hardware.
- IEEE Standard for Secure Computing Based on Trusted Execution Environment
    - This standard describes how computing systems and the developers of executables should act together. It describes standards and practices that the executable developers should follow to prevent security problems.
    - The Blender plugin ecosystem necessitates transparency for all 3rd party software. It is important that augmented software will not cause harm. For this reason, the execution environment standard is an extension of Blender's. A standalone executable is not an example of a viable final product because of the level of trust this would require.
- ISO/IEC/IEEE International Standard for Systems and Software Engineering - Recommended Practice for Architectural Description of Software-Intensive Systems
    - This standard describes how a detailed design document and other files should explain a complex software system. This is important for uniformity because software systems can have hundreds to thousands of components that are impossible for one person to understand.
    - The project will implement multiple software systems and libraries that will be implemented together creating a fully new process. Since the architecture of this project involves many moving parts it is important to follow practices so that the system is well documented. Individual parts should not break other parts of the system.

3 Project Plan

3.1 Project Management/Tracking Procedures

The project is best suited by the waterfall management style since the project consists of a concrete pipeline, where data is manipulated and outputted by each step. The linear approach of waterfall better suits this project design as other styles would introduce considerable difficulties(e.g. unable to test later steps in the pipeline without receiving input data from previous steps). Once a complete design is finished and reaches the maintenance or testing phases, the functionality of each component can be effectively assessed and any flaws will be addressed.

To track progress, the plan is to utilize gitlab issues to identify future steps and milestones in the project. This will allow for the identification of short-term and long-term goals for the project and stay on top of any deadlines to ensure a complete product by the end of the course.

3.2 Task Decomposition

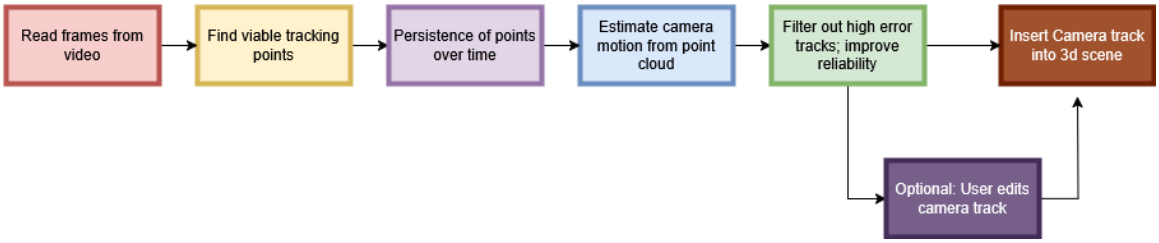

Figure 1. Task Decomposition Chart

The image above is a task decomposition chart. The chart shows how different processes of the project will be carried out. The task decomposition chart has been used to set out tasks for each team member to complete that allows for work to be evenly distributed between the group. It has also made it easy to break the project to be broken down into even smaller tasks.

3.3 Project Proposed Milestones, Metrics, and Evaluation Criteria

A background for some key milestones of the proposed project are reading frames from videos, finding viable tracking points, persistence of points over time, estimation of camera motion, filtering out errors, improving readability, and user UI interface. These milestones are important because they build on top of each other. For example it is needed to read frames from a video before performing calculations on points in the video.

1. Reading frames from videos: Program should be able to read in and have the ability to process each individual frame.
2. Finding viable tracking points: The program will be able to look across all of the frames and detect high contrast points to be used as tracking points. The user should be able to change the amount of points within the Blender UI.
3. Persistence of points over time: This is to make sure that the point tracker does not have large jumps over points that leave the screen, from camera jitter, or other color contrast problems. The plan is to get to 90% accuracy in calculating good persistence over time.
4. Estimation of camera motion: This is that calculated output that will be applied to the object being put within the video. This is directly affected by the accuracy of persistence of

points over time, so a 90% accuracy rate is also planned with the ability for the user to delete error points by themselves.

5. Filtering out errors: This will be available within the Blender UI plugin that gives the user an interface to delete certain points out that cause the final render to be jittery.
6. Improving Readability: Hopefully displaying the calculated data that the project outputs is displayed such that all users can understand. The ability to export to common file formats(CSV, JSON) can let users display that data however they want.
7. UI Interface: Once again within Blender it will allow new users to use the project without having a lot of background knowledge or being an expert in the field.

3.4 Project Timeline/Schedule

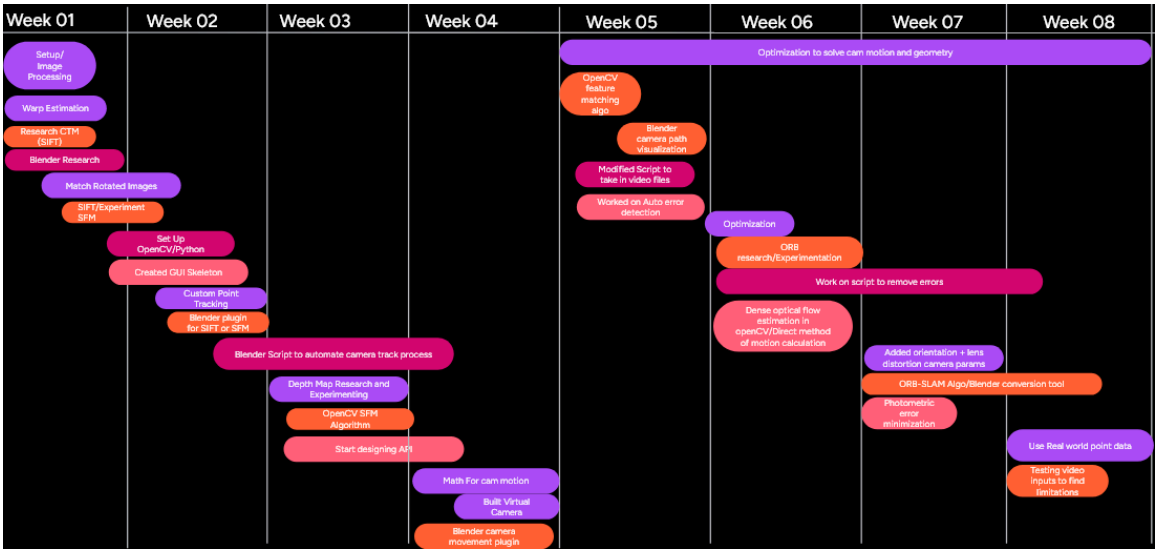

Figure 2. Gantt chart

Figure 2 shows a Gantt chart that outlines the major tasks completed over the course of the semester. Contained in the chart is a schedule that offers rough dates for the development of each aspect of the algorithm. This Gantt chart suits the waterfall model in the linearity of the approach, with each step of the project pipeline being addressed in sequence. The length of each line corresponding to each task roughly estimates the amount of effort necessary to complete each milestone on the schedule.

3.5 Risks and Risk Management/Mitigation

Persistence of points over time: It is likely that the point tracking system loses track of high-contrast points when the camera moves too quickly. This can be detected by observing the error as a result of each individual point. If high-error points exist, it may be necessary for the user to manually remove some points, but in most cases, the software should prune these points automatically and detect new ones.

Performance metrics: For the best user experience, this product should allow a beginner to create a camera track in under 15 minutes for a video that is less than 30 seconds in length. To mitigate the risk of not meeting this criteria, regular performance testing will be practiced while working on the design. After implementing new components, performance testing can be conducted to monitor

changes. This allows for the isolation of performance issues to specific systems which can be addressed by modifying the algorithm used or identifying alternate solutions to generate the necessary data.

Filtering out Errors: The ability to differentiate between error-prone tracking markers and successful tracking markers is a necessity. If the results of error-pruning are indeterminate, the solution will perform very poorly when moving subjects are in frame.

3.6 Personnel Effort Requirements

Task	Estimated person-hours	Description
Read Frames from Video	15	Setup a plugin for Blender and read in frames from video input on the Blender UI side.
Find Viable Tracking Points	10	Analyze video to identify high contrast points across frames.
Persistence of Points over Time	20	Create estimation of path of identified points throughout frames.
Estimate Camera Motion from Point Cloud	20	Use calculated paths to solve for the camera motion and output camera path in a usable format.
Filter out High-error tracks	12	Refine algorithm by identifying and filtering outliers and other paths that contain errors that can affect the final generated camera path.
Improve Readability	16	Design the UI in a user-friendly manner, aimed at casual users of Blender and this project.
Allow User Control of Camera Tracks	12	Add extra settings for more experienced users to tailor camera track to their needs while retaining the user-friendly design.

Table 1. Personnel Effort Requirements

3.7 Other Resource Requirements

Some resources needed to complete this project are any laptop as well as a camera more than likely a phone camera will be used. If needed, a camera that will perform better will be chosen. This project also requires certain software like Blender, Python, and OpenCV.

Blender itself will need to be installed by the user. Whereas the project will come packaged as a plugin product that the user will have to import into Blender.

A variety of cameras will be useful in the testing phase to ensure the algorithm works independent of the equipment used to capture the footage.

4 Design

4.1 Design Context

4.1.1 Broader Context

This product is primarily designed for two communities: The VFX artist community and the independent 3D artist community. The problem mainly affects both the welfare and economic areas of these communities. The main aim of the product is to provide an alternative to other free camera tracking solutions that is able to deliver professional-grade results without the immense cost that professional-grade camera tracking solutions come with. This could be immensely beneficial to the independent 3D artist community especially, as current solutions like SynthEyes or even After Effects come at prohibitively expensive costs due to these softwares targeting studio settings rather than independent artists. While the final product may not have the same vast suite of tools that software like SynthEyes provides, the purpose is to provide an easy-to-use and modifiable system to improve the Quality of Life of VFX artists through the camera tracking process in Blender.

4.1.2 Prior Work/Solutions

Blender Camera Tracking
The default camera tracking tools included with Blender are meant to provide a rudimentary way for artists to create camera tracks without having to use external software. It benefits from being fully integrated into the app itself, and produces acceptable results in a lot of scenes, but struggles with scenes which contain movement, water, or reflections. Users suffer from a tedious workflow that can be overwhelming to beginners. This is one of the main problems targeted to solve with the final product.

Meshroom
Meshroom is a software wrapper for the AliceVision photogrammetry pipeline. Can be used to generate a point cloud and camera track using the Structure from Motion algorithm. The main pro of this solution is its ability to generate a usable camera path without much user input, with relatively high accuracy. However, the Structure from Motion algorithm can take upwards of several hours to analyze and produce a result for a small video (<30 seconds). This is impractical for the purposes of the proposed requirements, as the target audience of this software would expect to produce a usable result in under 15 minutes for the purposes of quick prototyping.

SynthEyes
SynthEyes is an advanced tracking tool meant for industry VFX work by professional users. Contains a suite of tools and alternative tracking methods, like plane tracking. This is great for professional VFX artists performing studio work however its price ($62 per month as of May 2025) is prohibitively expensive for independent artists. SynthEyes is not intended for the casual user as this proposed product is.

4.1.3 Technical Complexity

The primary components of the proposed camera tracking system are the point tracker, the camera motion solver, and the Blender plugin.

The camera tracking system is a sophisticated piece of software. The first attempt used a custom-made point tracking system which works differently than any existing approaches. It used a machine learning algorithm called Triplet Loss, which is a relatively new engineering technique in computer vision to map images to points in space such that similar images occupy similar locations. With this, a neural network was trained to map points on an image distorted by a perspective to those on an image of the same subject under a different distortion. This required knowledge of machine learning, namely, auto encoders, embedding vectors, and the pytorch library.
Unlike existing solutions, which only consider sparse, high-contrast points, this solution used the texture of segments from a video. Using more information than points alone makes it possible to reconstruct perspectives more reliably under poor lighting conditions–an advantage over existing techniques.

The camera motion solving algorithm required extensive knowledge of linear algebra and calculus. Linear algebra was used to model a virtual camera which projects 3D points onto a 2D plane with perspective. It was also used to reduce some parts of the equations into matrix form to improve performance in Python. Calculus was used to take the derivative of the inverse of the camera projection equation and solve it with an optimization algorithm.

The Blender plugin component of this project includes the UI as well as code to translate the results from the camera motion solver to usable data. This requires knowledge of linear algebra as well as quaternions.

4.2 Design Exploration

4.2.1 Design Decisions

Integrating this project's algorithm as a Blender plugin was part of the initial design decision. When delivering a final plugin product that integrates seamlessly as a part of Blender, the plugin will hopefully be adopted quicker. When searching for users, an existing group—the Blender community—can be leveraged.

The most important part of the technical design decision was selecting the algorithm for reconstructing the scene in 3D. Although the customers do not care about the technical process, this decision was critical because the wrong choice could have been too inflexible. A decision too dissimilar from other ideas would have resulted in a burdensome time-cost for switching to a new

approach in the event of failure. Point tracking was chosen as the basis for the algorithm, similar to existing camera tracking algorithms. This decision allows for easy conversion to line or planar tracking if more scene geometry information is required.

The selection of Python as the language of choice was also a fundamental design decision. This choice is important because it impacts all phases of development. The decision was based on Blender using native Python code for plugins and the ideology of prioritizing productivity over performance to enable quick prototyping and optimization later. The thought is that a working prototype can be quickly integrated, with use of Python libraries, then transferred to a compiled language if needed.

4.2.2 Ideation

In order to solve for camera motion, a computer first must map out the geometry of the scene in the video. For this design decision, it was decided to go with the tracking-point technique, where small, high-contrast points are tracked as they move across the screen. Tracking points were chosen because it is a common solution to various types of vision-based tracking problems, and so existing algorithms are well documented. The mathematics also scale well in case later on it is decided to track lines instead of points, making it the most flexible option.

When brainstorming possible solutions for the design problem, other computer vision problems and the techniques used were looked at. Commonly applied techniques were likely to be robust and well documented, so were worth considering. A lot of consideration was put into innovative solutions. Tracking edges instead of points is a novel idea, but it is not so original as to be outlandish and unwise. It strikes a balance between creativity and proof of effectiveness, which is the standard for decision making.

Other solutions that were considered as follows:

- Line detention, which offers more information of perspective, but is more complicated.
- Pattern matching with a neural network, which was prototyped. It was robust, but too slow.
- Depth maps, which estimates a z-coordinate for each pixel and may be able to help reconstruct geometry. After realizing that gradient descent worked great for reconstructing depth from parallax, this became redundant.
- Plane tracking, which is similar to line tracking but extended by a dimension.
- A neural-network-based approach that doesn't map the environment, but tries to estimate motion directly was considered, but there is no evidence that this would work, so the risky investment of time by building a prototype did not make sense.

4.2.3 Decision-Making and Trade-Off

The process used to identify the pros and cons or trade-offs between each of ideated options was initially based on what group members would think others would and would not use. Then other products that were similar to this proposed design were researched plus the corresponding pros and cons of those products. The graphic below is a Product Research table, used to find pros and cons of the other products.

Product Services and Design *What is the product?*	**Unique Value Proposition** *What makes this product unique?*	**Product Advantages** *What are the things that provide a leg up?*	**Product Disadvantages** *Where might drawbacks exist?*	**User Pros** *What do users like about the product?*	**User Cons** *What do users NOT like about the product?*
Meshroom 	-Free software that provides a GUI interface where you can customize the photogrammetric pipeline to suit your needs. - Can generate colored points clouds	-Able to generate a path without much user input. -Generates point cloud, useful in aligning camera path to object.	-Can struggle without user inputting camera focal length. -Very slow to generate camera track. -Can't interact directly with Blender.	-"Works very well with rough textured objects." -"Great free solution that is easy to use and yields decent results."	-"Requires more dedication on the user's part than commercial solutions" -"Struggles with scenes containing shiny objects"
Kdenlive 	Free and open source software that allows for users to create and or import external plugins. Has motion tracking feature but nothing built in specifically for camera tracking.	Because of the diverse plugin support many effects can be transferred over to KdenLive.	-Has a small team of developers, slow to fix problems. -Does not support GPU acceleration for editing effects and playback, just for encoding during rendering.	-It is free and open source. -Not bloated and just focuses on its main function. -Easy to use for beginners.	-Reliance on external libraries can cause dependency issues. -Documentation is not as good as the most well known video softwares.
Adobe After Effects 	-The 3D camera tracker analyzes video sequences to extract camera motion and 3D scene data. The 3D camera motion allows you to correctly composite 3D elements over your 2D footage.	- Easy to use - Because it is built in to After Effects, compositing simple 3D elements is easy	- Lower quality camera track (high error) - User has no control over the tracking process. If the algorithm fails to solve for motion, there is not much the user can do to fix it. - Inextricably built into the After Effects software and cannot be used on it's own	-Wide range of features for animation, compositing, and visual effects. -Good integration with other adobe features.	- Mediocre results, not ideal for VFX shots that require precision. -Big learning curve. Since it has a lot of features and capabilities it can be too much for users at times.
Syntheyes 	- Offers alternative tracking mechanisms, such as plane tracking in addition to point tracking - Gives the user complete control and very advanced tools	- Offers a suit of advanced tools - The user has complete control over the processes	- Complex UI - Expensive	- Can result in very high quality tracks from all kinds of footage - Viable in industry	- The complex UI makes for a steep learning curve

Table 2. Product Research

4.3 Proposed Design

4.3.1 Overview

The current design will be implemented as a plugin within Blender. The main components of this plugin is the UI and the backend code that runs calculations on the video feed. The UI plugin will feature input boxes and sliders that change how the backend calculations will be done. The backend code will feature algorithms that choose points, take the user inputs, calculate the camera movement, and then apply that movement to the object in the scene. All of the user interaction will be with the Blender GUI through a custom UI.

4.3.2 Detailed Design and Visual

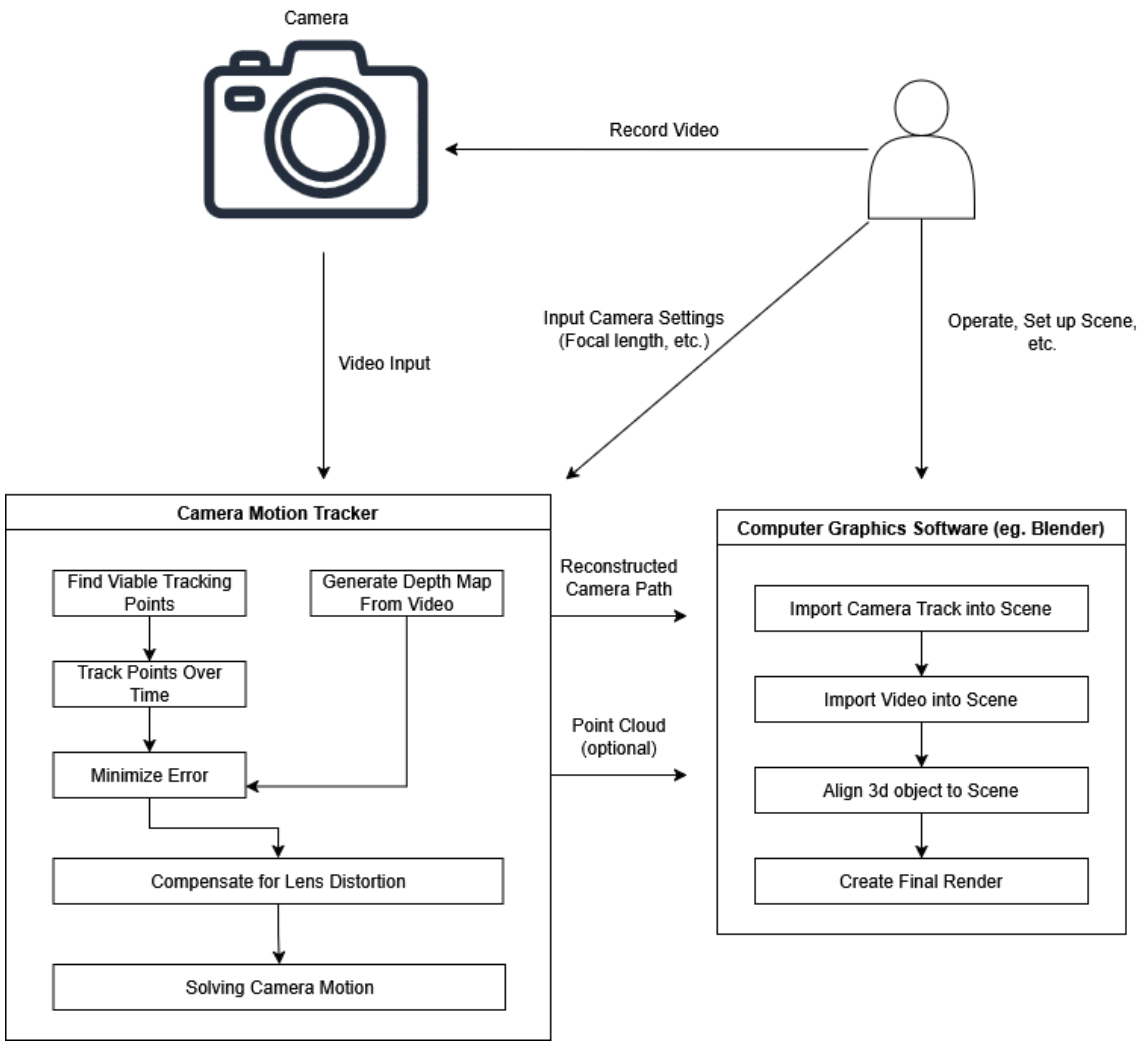

Figure 3. Detailed Design Diagram

Camera: The Camera is used to capture the video to be used as input for the Motion Tracker.

Camera Motion Tracker: the software to take a video and reconstruct the path of the camera used to film it.

- Find Viable Tracking Points: Analyze the video to find points that can be easily tracked across frames.
- Generate Depth Map From Video: Analyze video to estimate the depth of points from the camera lens.
- Track Points Over Time: Using previous identified points, generate tracks tracing the path of points as they move throughout the video.
- Minimize Error: Using these paths in conjunction with the depth map, eliminate outliers from the previous tracked paths.

- Compensate for Lens Distortion: Using camera metadata and/or user input, increase the accuracy of tracked paths according to lens distortion.
- Solving Camera Motion: Using the final set of tracked paths, compute the motion of the camera used to film the video in 3D space.

Computer Graphics Software: Software to render 3D graphics.

- Import Camera Track into Scene: Take path from the camera reconstruction algorithm and import it into a 3D scene.
- Import Video into Scene: Set video from Camera as the background of the animated camera path.
- Align 3D Object to scene: Set 3D object within 3D scene to be rendered on the 2D video.
- Create Final Render: Generate final video with 3D objects accurately superimposed on 2D video.

User: Films video to be used in the tracking software, inputs camera settings, tweaks camera path if necessary, and operates CG software to create the final render.

4.3.3 Functionality

To be accessible for the beginner users, this software product aims to have a simple user interface. The user is required to upload a video, which the algorithm then uses to calculate the camera motion. At this point, Experienced users can choose to tweak the resulting camera path according to their needs. The Blender plugin then imports the resulting camera path into the 3D scene, setting the user submitted video as the background. From this point, the user can align their 3D object(s) to the scene and create their final render within Blender.

4.3.4 Areas of Concern and Development

The current design is ideal for the casual user. It presents a simple user interface so as not to overwhelm beginners. The UI is designed to be minimal and easy to understand on the surface, hiding the complicated options from view by default. Simplicity is an effective tool in satisfying the criteria of a shallow learning curve.

The first primary concern is with the performance of the software. The software is meant to give quick and accurate results, ideally in 15 minutes or less. Longer waiting times are likely to frustrate the user or drive them to use other software solutions.

A second primary concern is regarding a professional user base. Design plans include providing an optional suite of settings for the professional user to tweak and modify the result to meet specific needs. However, extensive industry experience with professional VFX software is lacking. As a result, there is a high possibility that aspects of the process may be neglected, which could frustrate a professional user.

To address the first concern it is planned to do regular and rigorous performance testing to ensure that the software meets specified performance requirements. It is also planned to conduct surveys and user based testing to obtain user feedback from both casual and professional users to identify areas of improvement and the overall usability of the software so as to satisfy both audiences with the final product.

4.4 Technology Considerations

The first distinct technology that is being used within the design is the free and open source software Blender. Blender has a large and loyal following of both professional and amature users. Blender also does receive updates at a relatively consistent rate, showcasing how it is here to stay. Though the weakness of constant updates is that it could render the codebase of the final plugin product obsolete if changes are not upkept. The reason Blender was chosen is because it has lots of tutorials and great documentation online for interfacing with a UI plugin. Blender features many libraries that will make the reusability of the code much easier.

The second distinct technology that is being used is OpenCV for Python. OpenCV implements many complex algorithms that can and will be used within this project. The OpenCV library is periodically updated, though not as often as Blender. The OpenCV library tries to keep backwards compatibility between small changes, but on massive version changes things can break. The strengths of implementing features from the OpenCV library is the efficient implementations with C-Python. The C-Python optimized Python functions to use C and inline assembly for specific parts of the code to ultimately reduce runtime. This directly correlates to the time requirements in the design considerations. A weakness of the OpenCV library is that some important details are hidden and therefore it is much harder to understand the processes underneath the hood. If the design requirements require a fully customizable algorithm, it will need to be written from scratch.

4.5 Design Analysis

Part of the UI for the Blender plugin has been built, and work has begun on the backend code required for calculating camera movement. So far, the proposed design has functioned as a rough prototype. One issue encountered is the speed of calculations. Implementations of the algorithms take longer than expected. It is believed that faster, more error-prone algorithms could be used and run through an error detection algorithm, potentially speeding up the computation.

The program has worked seamlessly with the Blender plugin, and no problems are expected going forward. The primary obstacles delaying progress are the speed and accuracy of the currently designed algorithm.

5 Testing

5.1 Unit Testing

When developing the optimization algorithms, namely ADAM and gradient descent with momentum, a dummy function was used to make sure they were implemented correctly and the hyperparameters were well tuned. Finding the roots of a reasonably complex function such as a high-degree polynomial is similar to the camera path solving. Ensuring that the correct roots are returned was a way to test this isolated piece of code. This was easy to measure objectively. It was simply verified that the results were near enough to the known roots.

The virtual camera was another piece of code that could be easily isolated, and therefore tested. The coordinates of a cube were generated to make sure it was able to successfully render a 3D wireframe cube. Unfortunately, no overall objective measure for correctness was known, but there were a few standards that could ensure correctness was met. For example, points farther from the camera should be moving more slowly when the cube moves because of parallax. At first, the results did not

look right, and this test showed that a single negative sign was missing in the projection math, as parallax had the reverse effect.

5.2 Interface Testing

There are two main user interfaces in the application. The first is the setup interface, where the user submits their video, adjusts various input settings(e.g. focal length). The second interface allows users to make edits to the generated camera path if desired, and import the camera path into the 3D scene when ready. The functionality of all of these features can be tested. Blender comes with a framework that allows developers to use Python to fire off different events regarding UI (ui-simulate), in a similar fashion to tools like Selenium. While using this framework, accurate functionality and flow of data for the interfaces can be ensured.

5.3 Integration Testing

The critical integration path in the design is the connection between the different stages of the algorithm. This is important because each stage depends on the stage that came before as input. When making changes to one stage it could make large changes to the output. For this reason it is important to test frequently throughout the building process and make sure past features don't break.

These different stages will be tested by individually testing the components with known input and outputs, making sure it is the correct data type. PyTest is a library that can be used to make integration testing for Python. This library takes each individual function in the project and tests the component by itself. When new features are added and or removed, integration testing should be run such that all tests run correctly. For entirely new functions, an integration testing function must also be built testing the new function.

5.4 System Testing

Because all of the tests use Python libraries to test units and interfaces, it is possible to create a comprehensive Python script that combines tools like PyTest and Blender's ui_simulate to simulate the workflow of a typical user of this product. By simulating the whole workflow the connection of different systems will be observed, and the utilization of logging will ensure accurate data flow throughout each step of the pipeline, as well as the total time it takes to generate a camera path after inputting a video. These scripts can be tested with various videos under different conditions(e.g. time of day, amount of movement) to see how the software handles different scenarios.

5.5 Regression Testing

With the addition of new features to the codebase, performance requirements must be upheld. The software is designed to enable both casual and professional users to rapidly prototype various shots or generate accurate camera paths for final renders. To meet this objective, results must be produced in under 15 minutes to prevent user frustration. An alternative option that offers higher accuracy at the expense of time may be acceptable.

New features should be tested on the same hardware to verify that the software functions effectively on any system meeting Blender's minimum requirements. Interface changes must avoid introducing unnecessary complexity that could alienate casual users. Algorithmic updates should maintain the

same input and output structure unless a change is strictly necessary, in order to preserve compatibility across modules. Existing tests for unmodified systems should be executed to detect any unintended behavioral changes resulting from new additions.

5.6 Acceptance Testing

Demonstration of the design requirements, both functional and non-functional, will be accomplished through thorough documentation of all features and clear explanations of their usage. During the demonstration, a walkthrough of the algorithm's execution will be provided, with detailed descriptions of its behavior. Client involvement will occur in two ways: direct interaction with the product, including the opportunity to ask questions based on specific goals, and task-based testing, where the client is given a predefined objective to complete using the system. This process provides valuable feedback on the usability and clarity of the user interface and documentation.

5.7 Security Testing

Security testing is relatively unnecessary for this project. The software is written entirely within Python and has zero internet connectivity. Common security vulnerabilities regarding network communication or unsafe memory handling do not apply to this project.

5.8 User Testing

As the code is open source, participation in Blender communities on messaging platforms such as Discord will enable outreach to individuals outside of the development group to test the product. This approach ensures that testers are genuinely interested and have real use cases for the project, providing valuable and applicable feedback.

Given the lack of personal connection between testers and developers, the expectation is for more candid and realistic feedback. Members of the Blender communities can offer insight into whether the project integrates effectively into existing workflows. Feedback will guide adjustments to the project, potentially including UI enhancements and requests for greater customizability.

An additional benefit of open source development is the possibility of community-driven contributions. If a member of the Blender community submits a valuable improvement and it gains positive reception, inclusion of the changes into the official release becomes a viable option. This fosters a sense of ownership among users and encourages wider adoption.

5.9 Results

The gradient-descent based approach easily gives measurable results, using the loss function as a precision metric. The error, shown in the right column, is the squared sum of the difference between estimated and the actual projection position of the points and the camera sensor. A good camera track has an error of less than 1, which is achieved after running the algorithm for 300 iterations. Because of the success these results show, work continued on this approach. Ideally, the same error would be achieved with fewer iterations.

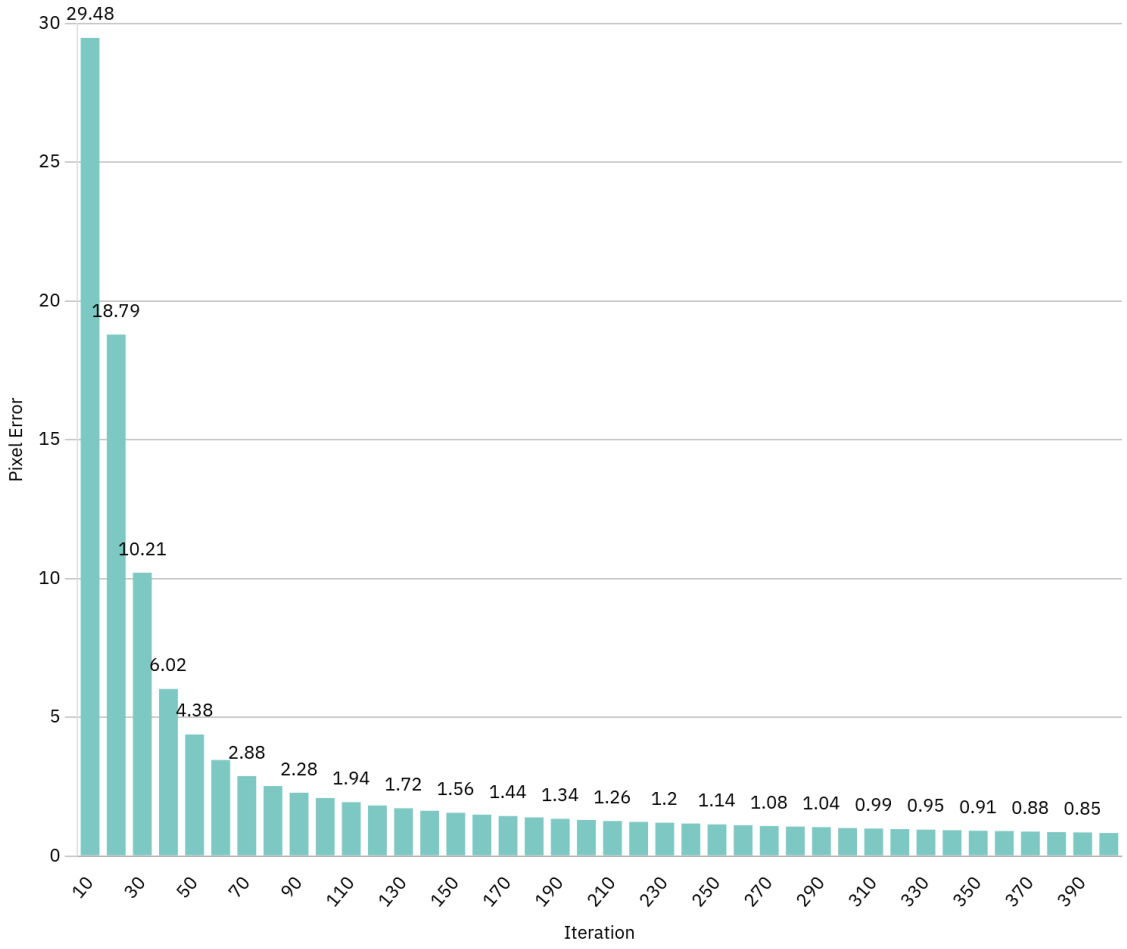

Figure 4. ADAM optimizer test results on virtual scene

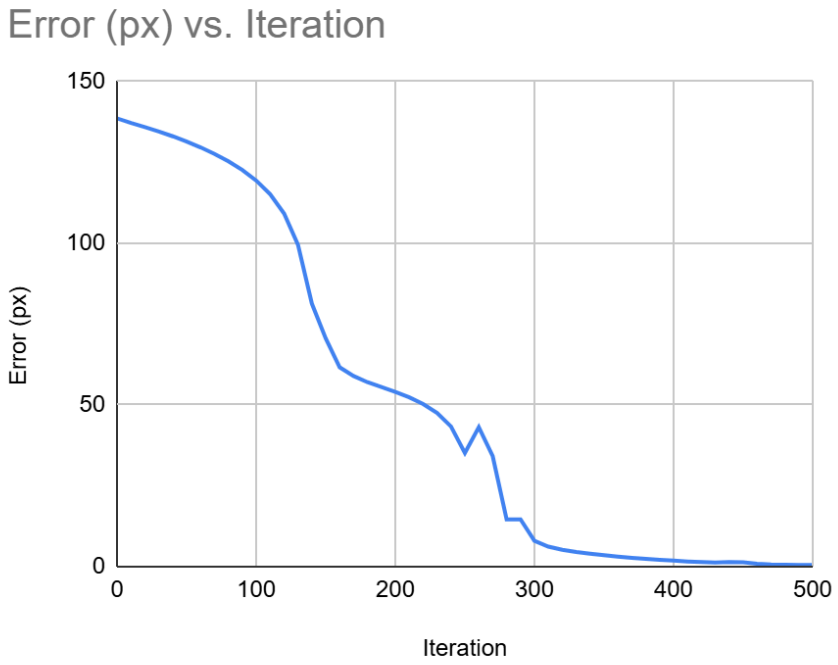

Figure 5. ADAM optimizer test results on real scene

6 Implementation

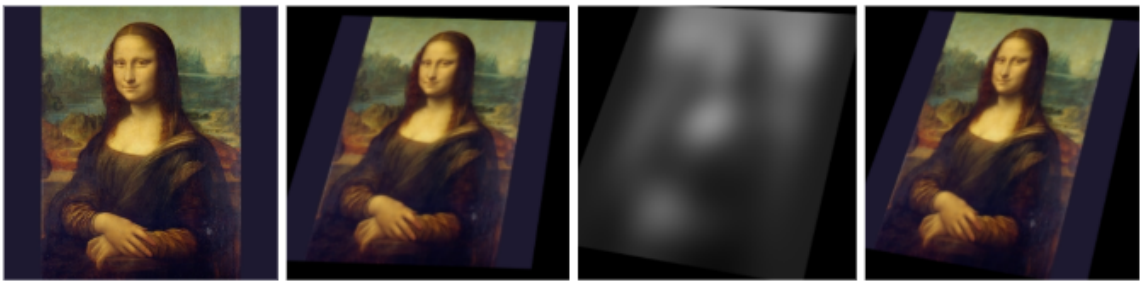

Figure 6. Estimating image distortion with optimization

The first attempt at a camera motion solving algorithm was created with the intention of avoiding tracking points, and instead estimated the distortion from perspective shifts on entire planes. Figure 6 shows the results of the implementation. The leftmost image is the original image. The original is distorted to simulate a perspective shift, resulting in the second image. Next, the image is blurred so that pixel information in one region in space is carried over to other regions (this is the definition of a blur). As a consequence, the optimizer may know which direction to step towards to repeat the transformation even when the current position in the transformation space is far from the solution. Finally, the original image is blurred, and a series of small distortions is applied until it resembles the 3rd image. Through this series of steps, the perspective distortion of the second is

discovered and displayed in the fourth image. The 2nd and 4th image should be identical. Under close scrutiny, it is apparent that they are slightly different. The algorithm failed.

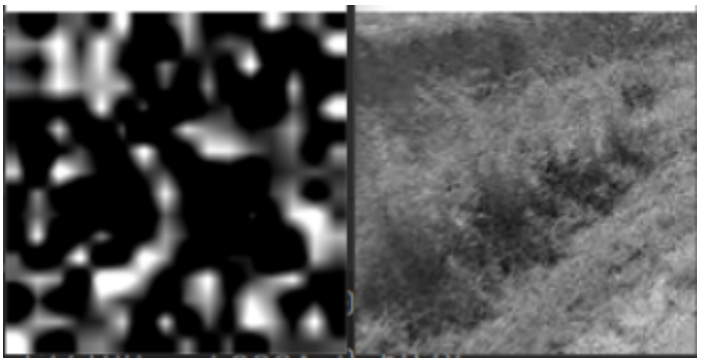

Figure 7. Image partition and its vector mapping

The next attempt to solve for the distortion of planes is shown above in figure 7. An image is broken down into many sub-section. One such subsection is shown on the right. A neural network was trained and used to determine if two images are of the same subject under different rotational transformations. This is done by mapping the image to a vector and comparing it to the vector mapping of a rotated image (the visualization of the vector corresponding to the right section is shown on the left). If an image partition is small enough, the perspective distortion becomes negligible and the only effective distortions are rotation, scale, and position. Therefore, if the neural network produces similar vectors given two image sections, each from images with different perspective distortions, then the points in space which the two image sections depict must be the same. After creating numerous segment mappings from an undistorted image to a distorted one, the perspective distortion of the image as a whole can be solved. The result of this technique is shown below. The green lines in figure 8 connect pixels in the current frame with pixels in the previous frame. In other words, it estimates the pixels' movement.

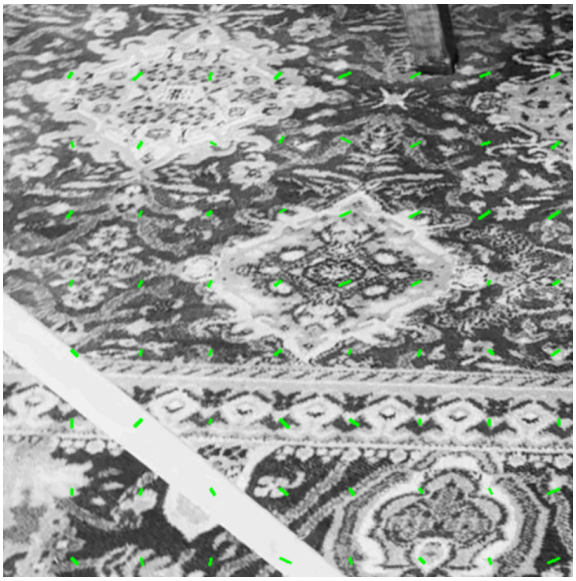

Figure 8. The estimated motion direction of pixels in a video

Figure 9. The movement of vertices on screen due to camera motion

The latest approach solves for geometry and camera motion at the same time using the ADAM algorithm. This is done by taking the gradient of the error between the tracking points on screen and the projected points that result from the predicted camera position and the scene geometry. By taking small steps in the opposite direction of the gradient, the error will eventually reach zero. The blue lines in the figure show the motion of the rendered cube due to arbitrary camera motion. The red lines show the motion of the same vertices due to camera motion that was reconstructed by the algorithm. The two sets of lines are nearly identical, indicating that this approach works well. Below, figure 10 shows the same technique on real footage.

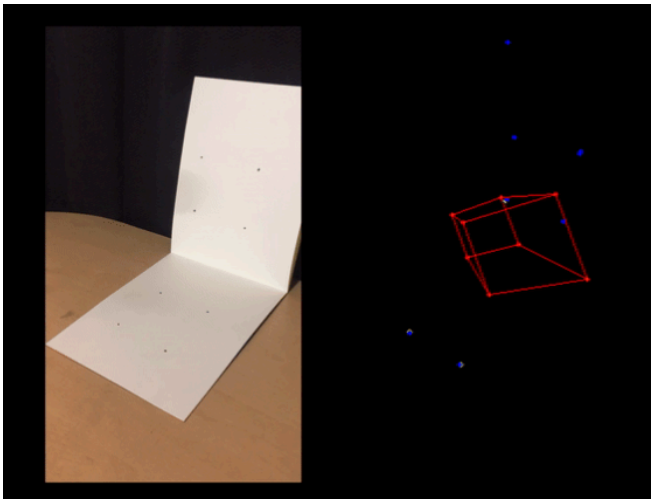

Figure 10. Motion solving via optimization on real-world data

7 Ethics and Professional Responsibility

7.1 Areas of Professional Responsibility/Codes of Ethics

Area of Responsibility	**Definition**	**Software Engineer CoE Equivalent**	**In This Project**
Work Competence	The product should be of high quality, and be developed with integrity.	Product - *"Software engineers shall ensure that their products and related modifications meet the highest professional standards possible."*	Keeping the software up to high standards, to ensure a very positive user experience.
Financial Responsibility	The product should deliver good value without incurring unnecessary costs.	Client and Employer - *"Software engineers shall act in a manner that is in the best interests of their client and employer consistent with the public interest."*	No current cost in the development of the project so far, and plan to make the final version open-source.
Communication Honesty	Transparency will be maintained regarding all information about the product and its development.	Public - *"Be fair and avoid deception in all statements, particularly public ones, concerning*	In each weekly development report, it is clearly communicated the development process

		software or related documents, methods and tools."	of this software, ensuring transparency.
Health, Safety, Well-Being	Minimization of safety risks to all stakeholders will be considered whenever possible.	Public - *"Cooperate in efforts to address matters of grave public concern caused by software, its installation, maintenance, support or documentation."*	It is ensured that the product does not pose any risk to users thus far. This is since the software runs entirely on a user's machine without need for network communication.
Property Ownership	Use of others' work or intellectual property will occur only with explicit permission and proper credit.	Management - "*Ensure that there is a fair agreement concerning ownership of any software, processes, research, writing, or other intellectual property to which a software engineer has contributed."*	The product does not incorporate others' work without credit, and proper permission will be obtained when necessary.
Sustainability	The product should not pose any threat to the environment or natural resources.	Public - *"Approve software only if they have a well-founded belief that it is safe, meets specifications, passes appropriate tests, and does not diminish quality of life, diminish privacy or harm the environment. The ultimate effect of the work should be to the public good."*	The product does not directly affect the environment in any meaningful way.
Social Responsibility	The product should produce a benefit for society.	Public - *"Software engineers shall act consistently with the public interest."*	The product is intended to simplify the vfx workflow, saving artists time and providing better quality of life.

Table 3. Code of Ethics

The team is performing well in the area of Social Responsibility. One of the main motivations behind the product is to make VFX more accessible to casual users as well as independent

professionals who do not have the budget to use industry standard software. The aim is to improve these users Quality of Life by eliminating one of the most tedious aspects of the VFX workflow within Blender. The aim is also to make the final product free and open-source, to provide a quality alternative to other VFX solutions, which can be quite expensive.

An area which may need improvement is Health, Safety, and Well-being. This project's software codebase cannot pose any physical risk to users, but there is always the risk of potentially insecure or unsafe code. While the product operates on the users' local network and is written entirely within Python, which protects from many vulnerabilities that could be exploited over the network or through a language that doesn't use a garbage collector, it does not rule these things out completely. Thus, special care should be taken to ensure the codebase is free of such vulnerabilities to ensure a safe, quality product for all users.

7.2 Four Principles

Context Area	**Beneficence**	**Nonmaleficence**	**Respect for Autonomy**	**Justice**
Public Health, Safety, and Welfare	The product could increase quality of life by eliminating certain tedious workflows.	The product does not pose any harm to any user.	The software is easy to understand so all users can make their own decisions.	The product is designed for both professional and casual users.
Global, Cultural, and Social	The product enables non-professional users to do their own VFX work.	It's possible but unlikely for the product to eliminate jobs. Could possibly create more for less technical users.	Users are able to choose the level of control they have over the final result.	The product simplifies the process for casual users unfamiliar with many VFX principles.
Environmental	The product will not directly benefit the environment.	The product could not harm the environment in any way.	The product does not affect the environment in this way.	The product does not affect the environment in this way.
Economic	The product provides a free alternative to expensive vfx solutions.	The product will not use any exploitative economic practices, it will be open source.	The product does not seek to replace any other completely; users are free to make their own decisions.	The product being freely available opens up opportunities for users who cannot afford more expensive solutions.

Table 4. Four Principles

Respect for Autonomy and global, cultural, plus social is a pair that is quite important to us. As this product is meant to appeal to both casual and professional users, it is important to take special care in the design decisions to work towards a product that is both easy to use and understand for the casual user, while also providing the optional but detailed controls to satisfy the professional user.

Nonmaleficence and global, cultural, plus social is an area that the product may currently be lacking in. The product is intended to allow more people to produce their own VFX work, not to eliminate jobs in VFX. It's highly unlikely that this would happen, as this product is aimed at an independent audience using tools common among this audience(e.g. Blender). A potential solution is to try and restrict the allowed use of the plugin software to independent usage only, but that would be quite difficult to enforce independently and could also alienate many professional users.

7.3 Virtues

Team Virtues

Honesty – Accuracy has been prioritized in the presentation of work to ensure a transparent overview of the product for all users. Continued vigilance in tracking decisions and progress will support clear and truthful communication regarding development efforts.

Responsibility – No actions will be taken that may cause harm, including the use of insecure code or violation of intellectual property rights. All components are original unless proper and explicit permission has been obtained. Care will also be taken to ensure that the software remains free from vulnerabilities.

Accountability – Weekly responsibilities have been completed in a timely manner and will continue to be managed through consistent reports and meetings. Reports are, and will remain, available on the team website. Regular meetings will persist to ensure progress remains on track.

Individual Virtues

Isaac – I have demonstrated honesty within my work so far. This is important to me not only because honesty promotes better communication but allows transparent feedback in the team's research and testing. By myself, I have worked on specific camera tracking ideas and sharing how effective they may be. Without being honest about how my idea might not be working effectively, the team could have been pushed behind.

Another virtue I think is important is curiosity. Attempting a novel approach to camera tracking more often than not could result in failure and "wasted" time. It is important to be curious about trying new implementations just to see the result, rather than thinking about the possible failure. I will demonstrate the curiosity virtue by not being afraid to think of new ideas and approaches to the camera tracking problem.

Will – I believe that I have demonstrated the virtue of rationality in my work so far. Much of my work this semester has been in researching and experimenting with existing algorithms that have the potential to suit this project's purposes. I have had to utilize the virtue to make judgments on these algorithms to ensure they suit and up to the proposed standards and expectations regarding performance and ease of use. I believe this virtue is important to design decisions to ensure a higher quality final product.

Another virtue I consider to be important is Justice, to ensure that whatever this product ends up being in produces a concrete benefit to society. I haven't demonstrated this virtue so far, but by bringing these considerations to mind while making design decisions will help bring out this virtue to ensure justice in my work.

Andrew – A virtue I have demonstrated thus far in this project is perseverance. This is important because throughout life you will be presented with challenges that will not be finished if you do not persevere. With perseverance you will be less likely to give up which should be important for everyone. I have demonstrated perseverance by working through the steep learning curve this project provides. This project has had many things each of us have had to learn to be beneficial for the team.

A virtue I think is important that I have not demonstrated in senior design is contentment. This is an important virtue because being happy with what you have done is good to have. I feel that I am not satisfied with what I have done because the project is not finished. I am not saying that I am not happy with what has been accomplished so far. I think this project could truly turn into something special. I really want to see what it will look like when the project is done..

Eric – An important part of the design process is recognizing when an idea doesn't work. After spending hours developing a unique solution, a better solution may be realized, rendering prior work useless. It takes humility to recognize a bad idea and move on to a new approach. Without the virtue of humility, it is easy to become attached to an idea because admitting its ineffectiveness means admitting its creator's fault. I have exercised this virtue while testing a multitude of ideas, many of which have failed. After each failure, I moved on to a new strategy and eventually found a successful one.

Another important virtue is orderliness. This practice is critical for making effective documentation, which can be used to communicate progress and ideas with team members. Documentation is also helpful for engineers who need to reference their own prior work to remember the effect of small details that get lost in the complexity of a project. I will practice the virtue of orderliness by documenting my work so that someone outside of the team could hypothetically copy the design process.

8 Closing Material

8.1 Conclusion

From the start of the semester, the primary goal for this project was to reduce the tedium of the camera tracking process with an easy to use plugin or standalone software that appeals to all users. The research and experimentation performed over the course of this semester has shown this to be an achievable goal. Throughout the semester various strategies of camera tracking have been researched. It was found that other fields of tech, especially robotics, use similar technology to achieve similar goals, like localizing a robot within 3D space. The algorithms used for these techniques, such as ORB-SLAM3, could possibly be reworked for the purposes of this project. The secondary goal of the project was to present the camera tracking software through an easy to use UI, simple enough for casual users to operate, while also providing the necessary control that professional users desire in camera tracking solutions. Several prototypes of this interface have been

made, exhibiting the ease-of-use quality, but lacking in advanced control features. These features can be implemented in the UI design when they are implemented in the camera tracking algorithm.

In future designs greater emphasis will be placed on the meeting of performance metrics. The target is less than 15 minutes from the time of user input to the final camera track generation. This is not a target that has been met as of yet in any present implementations. After meeting such requirements the product should be packaged as an open-source plugin that can be easily distributed on Blender Extensions. If time allows, efforts will be made to tackle the problem of complex scenes including water or erratic movement which typically trouble current camera tracking solutions. Achieving these goals will make up the bulk of the work for next semester, ultimately resulting in the final product.

8.2 References

C. Campos, R. Elvira, J. J. Gomez Rodriguez, J. M. M. Montiel, and J. D. Tardós, "ORB-SLAM3: An accurate open-source library for visual, visual–inertial, and multimap SLAM," *IEEE Transactions on Robotics*, vol. 37, no. 6, pp. 1874–1890, Dec. 2021, doi: 10.1109/TRO.2021.3075644.

M. Kennerley, "A Comparison of SIFT, SURF and ORB on OpenCV," *Medium*, May 21, 2021. [Online]. Available: https://mikhail-kennerley.medium.com/a-comparison-of-sift-surf-and-orb-on-opencv-59119b9ec3d0

8.3 Appendices

Tentative Blender UI plugins made according to the Blender 4.4 Python API - https://docs.blender.org/api/current/index.html

9 Team

9.1 Team Members

9.2 Required Skill Sets for Your Project

- C++/Python experience
- Blender tracking
- Blender Python API
- Blender Plugin Creation
- OpenCV/Computer Vision
- Linear Algebra
- Camera Experience

9.3 Skill Sets covered by the Team

- C++/Python experience - Will, Andrew, Eric, Isaac
- Blender tracking - Will, Andrew, Eric
- Blender Python API - Will, Andrew, Eric, Isaac
- Blender Plugin Creation - Will, Andrew, Eric, Isaac
- OpenCV/Computer Vision - Will, Eric, Isaac
- Linear Algebra - Will, Eric
- Camera Experience - Eric

9.4 Project Management Style Adopted by the team

The waterfall management style will be utilized, completing each respective task in phases before moving on to the next. Occasionally prototype work is produced for future project phases for testing purposes.

9.5 Initial Project Management Roles

Eric - Motion Estimation Solutions

Isaac - UI Interface + Automating Error Detection

Andrew - Process Automation + Automating Error Detection

Will - Camera Tracking Research/ UI tools and Development

9.6 Team Contract

Team Members

Andrew Gooding, Isaac Kenyon, Will Ernatt, Eric Wittrock

Team Procedures

Day, Time and Location of Team Meetings: Thursday 2pm - 3pm

Preferred Method of Communication:Discord

Decision Making Policy: Majority Vote

Procedures for Record Keeping: Use a shared document with a table of meeting times and discussions. The meeting can be pre-planned according to the table, and needed adjustments will be made if the scope of the discussion changes.

Participation Expectations

Expected individual attendance, punctuality, and participation at all team meetings: Try to attend all meetings but if something comes let the team know beforehand.

Expected level of responsibility for fulfilling team assignments, timelines, and deadlines: If the assignment gets completed, you have met the required level of responsibility.

Expected level of communication with other team members: Speak whenever needed and have something that you have worked on prepared to talk about at each meeting.

Expected level of commitment to team decisions and tasks: Each team member should voice their opinions. Throughout the class, each member will have designated roles. It is especially important for someone to share thoughts that pertain to their role.

Leadership

Leadership roles for each team member: To be determined based off of team assignments.

Strategies for supporting and guiding the work of all team members: Any codebase merging will be done at the meetings so everyone is familiarized with issues and progress of other team members while avoiding merge conflicts.

Strategies for recognizing the contributions of all team members: At the beginning of the weekly meetings, each team member will present what they have been working on.

Collaboration and Inclusion

Skills, expertise, and unique perspectives each team member brings to the team:

Eric Wittrock - C++/Python, Computer vision with OpenCV, image processing with Keras and pytorch, experience with 3D graphics mathematics, experience with visual effects in Blender, Blender's Python API, and cameras.

Isaac Kenyon - Python, Java, C and 2D image processing experience.

Andrew Gooding - Experience with Python, C, Java, minimum experience with image processing and 3D creation software.

Will Ernatt - C++/Python, Machine Learning w/ Python, Java, minimal experience with Blender. C++ graphics programming (OpenGL).

Strategies for encouraging and supporting contributions and ideas from all team members: Making sure each other's opinions are heard and taken into consideration.

Procedures for identifying and resolving collaboration or inclusion issues: See previous decision making policy. Everyone gets a vote.

Goal-Setting, Planning, and Execution

Team goals for this semester: Have a working prototype by the end of the semester.

Strategies for planning and assigning individual and team work: Taking into consideration each member's skills and at meetings create a list of tasks to be assigned.

Strategies for keeping on task: Create and update weekly checkpoints at each meeting.

Consequences for Not Adhering to Team Contract:

How will you handle infractions of any of the obligations of this team contract?: Any issues that arise will be discussed in a team meeting.

What will your team do if the infractions continue?: Reaching out to the instructors is the next step.

****

a) I participated in formulating the standards, roles, and procedures as stated in this contract.

b) I understand that I am obligated to abide by these terms and conditions.

c) I understand that if I do not abide by these terms and conditions, I will suffer the

consequences as stated in this contract.

1) Eric Wttrock_____DATE: 2/6/2025_____

2) Andrew Gooding_____ DATE: 2/6/2025_____

3) Isaac Kenyon_____ DATE: 2/6/2025_____

4) Will Ernatt_____ DATE: 2/6/2025_____