Sight tracking in Unity

Halse, Douglas
Karlsson, Mattias
Larsson, Johan
Persson, Hannes
Östmo, Marcus

Contents

1 Introduction

This software is developed to help study different behaviors in human-computer interaction. It keeps track of where the user looks, for how long and where the user is located in 3D space. All of this information is saved to disk with timestamps in CSV-files. The software is designed to be easily implemented to any Unity project, by only having to add one script to the Camera in Unity. The information from the files makes it possible to replay a sequence based on the data. The files contain the following information:

- **Sequential data**
  Which game objects the user have looked at, and for how long in sorted in order of occurerrence. (same object may appear several times)

- **Summarized data**
  Total time looked at each object. (one object will only appear once)

- **Looking at and positional data**
  The SightTracker will also record where you are and what you are looking at on each frame. This depends on a setting which will be explained later in the `Manual` section.

2 Before first use - IMPORTANT!

2.1 Faulty data

Do not start and/or stop SightTracker during the runtime of the scene, this forces the tracker to initialize and adds incorrect data to the CSV file. If you observe a "Starting..." in your CSV file, you have encountered this issue. If this issue has occurred, you have to manually exclude the starting time from further calculations based on the data or rerun the test again.

2.2 Unique object names

Do not name the Unity gameobjects identically, the SightTracker package differentiates objects based on name. If you have two objects of the same name, when looking at either of them the stopwatch would run continually as if it was one object. This error is hard to notice in the CSV file unless you pay careful attention when running the test.

2.3 Objects that block player view

The SightTracker has not been extensively tested, there might be issues with using the tracker together with a player model holding an object that blocks the view. The SightTracker shoots out a beam and records the object that is hit, and certain player models might be in the way of the beam. If this is the case, it might work if you move the camera to a slightly different position in the player model.

2.4 Unity Keycodes to specify keybinds

You must use Unity keycodes when selecting keybindings for the debug HUD, keycodes can be found on Unity's website [1].

2.5 More than one scene

The SightTracker package has only been tested while using a single scene. When running a program that changes scene there might (most likely will) be undefined behaviour. The recommendation is therefore that you use this only if you are running a single scene.

3 Manual

3.1 Installation

This is what it looks like in `Unity 2019.3.10f1`. It might vary.

- **Importing to Unity**
  1. Open up your project, go to 'Assets' in the top menu bar.

  2. In the drop-down menu select 'Import package' and then 'Custom package'.

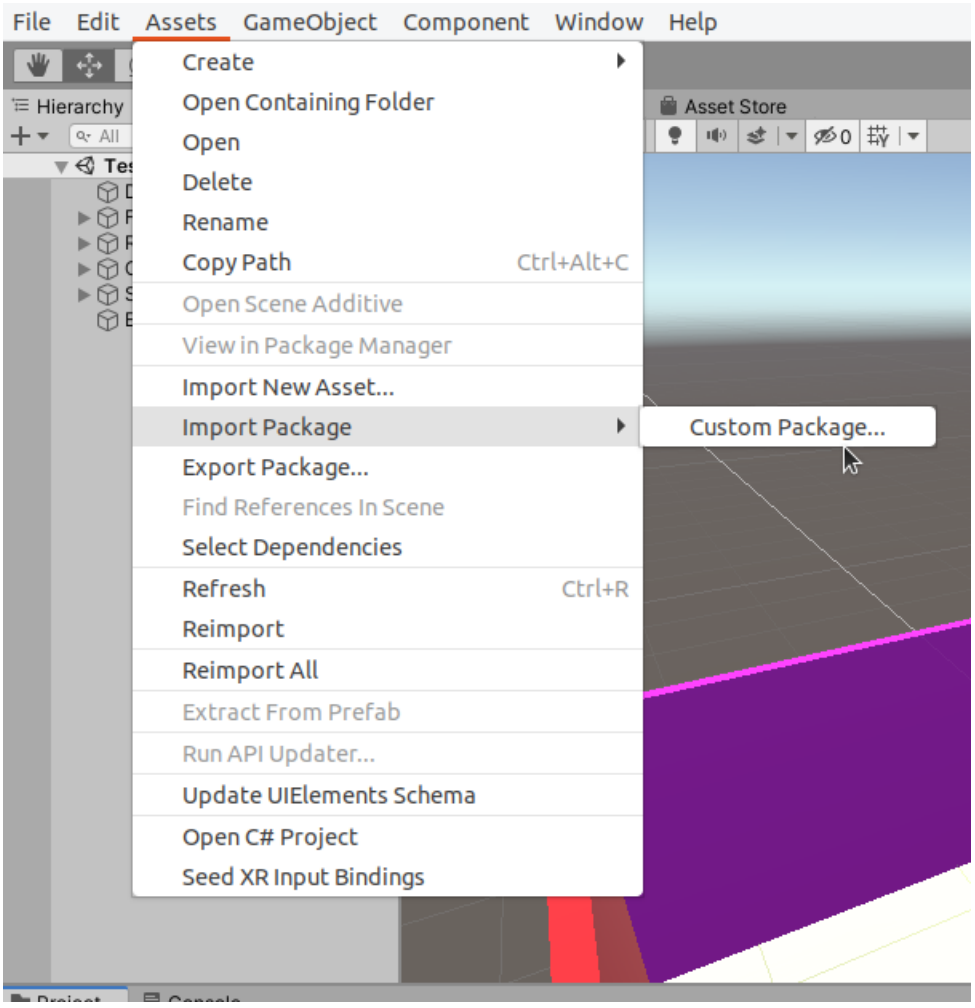

Figure 1: Importing a package in unity

  3. Locate the SightTracker package and select it.

4. You should now see the SightTrackerScripts folder in your project, open the folder and drag the script named SightTracker to the camera.

- **Settings**

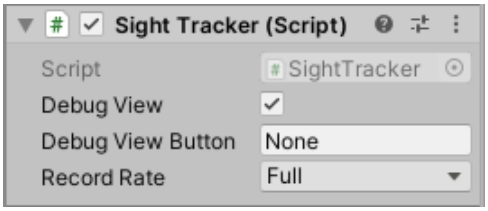

Figure 2: SightTracker has been added to the camera

5. The figure above is what the Inspector of the camera should look like. From here you can choose a key to toggle the debug HUD of the SightTracker. The key must, however, be a Unity Keycode [1]. From here you can also select the sampling rate. The options are: Full, Half and Quarter. This means that the Sight Tracker either captures each frame, every other frame or every fourth frame.

6. The SightTracker will generate three different CSV files on each run of the program, with timestamps to help distinguish each usage.

3.1.1 CSV output

The SightTracker package generates three different CSV files as mentioned earlier

- **CSVSequential** [Object, time]
  Contains a list of which objects the user have been looking at in order of occurrence and a duration for how long the object was looked at.
- **CSVSummary** [Object, time]
  Contains a summary of all objects looked at and the total time looked at each object.
- **CSVTimeData** [3D Vector, 3D Vector, TimeStamp]
  Contains two vectors, the first vector is where the player is looking, the second is the location of the user and the third is the time from start that the user is at this position and looks this direction.

3.2 Using SightTracker

As long as the script was loaded properly onto your own Unity project and attached, as a script, to the camera it should automatically generate the aforementioned CSV files.
The output of the script is to the folder `'/path_to_your_project/Sight_tracker/'`.
The Record Rate setting is essentially the polling rate of how often it records the users position and the direction of the camera. By setting this to `'Full'` the program records **every** frame which generates a huge CSV file if program runs for an extended period of time.

`'Full'` - Records the camera direction and position of every frame.
`'Half'` - Records the camera direction and position of every other frame.
`'Quarter'` - Records the camera direction and position every fourth frame.

3.3 Exiting SightTracker

For the recorded data to be saved in files on disk, the program must be stopped while in the same scene. If the scene is changed before exiting, no data will be saved to disk or at the very least undefined behaviour will follow.

4 Graphical representation of data

Since this software records a lot of data of what the user is looking at it would be nice to have a way to display the results. This could for example be using a pie or bar chart to show how long each object in the scene have been looked at by the user. Another way to display the collected data could be done by making a 3D heatmap over the entire scene.

As could be imagined, there are a lot of ways to display the data visually, and different users would most likely want it displayed in different ways.

Thus, we decided not to create an explicit program to display the data. Instead we are going to share some good resources for visualizing a CSV file.

If the user wants to create some diagrams for the data, the library `bokeh` [2] is a great option. This library can create interactive visualization graphs for web browsers. With a minimum amount of python code you could make the following bar chart from the SightTracker generated CSV data files. This chart can be viewed in any web browser and have features such as move, zoom and save.

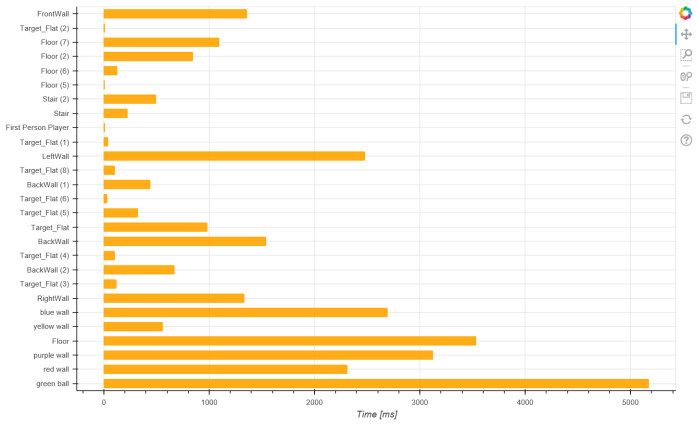

Figure 3: Visualization of recorded data

Using a bar chart makes it easy to see which objects the player was looking at the most/least. Depending on what the user wants to study this could be a good way of visualizing data. Of course, `bokeh` [2] is not the only option when it comes to visualization of data. For instance, another resource that could be used is the python library `matplotlib` [3].

5 What we would have wanted to add

- **Cross scene data gathering**
  The package would have supported several scenes and the loading in between them if we had the time.

- **Folder structure for CSV files**
  At the moment all CSV files end up in the same folder which makes it rather time consuming to find the specific file you are looking for.
  `'SightTracker/SceneName/DateAndTime/CSVFiles/'` would have been easier to browse.

- **Fully functioning replays**
  At the time we have a ReplayFromCSV script that works for our scene. We have noticed that it, however, highly depends on camera implementation of the user program. This is something we would have liked to solve.

- **Interpolation of replays**
  We also would have liked to implement some sort of interpolations if the user records in `'Half'` or `'Quarter'` speed. Reduce the amount of lines written to disk while still experiencing a seemless playback from CSV.

References

[1] Unity Keycode, https://docs.unity3d.com/ScriptReference/KeyCode.html

[2] bokeh, https://docs.bokeh.org/en/latest/index.html

[3] matplotlib, https://matplotlib.org/