Individual Assignment: Final Project

Deadline: **March 21st, 2021 at 23:59**
(prototype description + video)

The **final project** of the Augmented Reality course must be completed ***individually*** and will be evaluated based on your submitted prototype description and video. When explaining your implementation in the report, make sure to include **code snippets** to demonstrate your solution.

In this assignment you will first use your own camera intrinsics and OpenCV functions for image warping and unwarping (**homography**). Then, you will combine what you have learned so far to implement an **AR interaction technique**.

1. Homography

TASKS

a) Print one of the unwrapped texture templates (e.g., dog texture) and set it as image target in a new Unity scene. Display the corresponding model (e.g., dog model) on top of the image target. Use colored pencils or markers to color the white parts of the unwrapped texture. Define a button or key stroke to set the texture of the 3D model to your customized printed template. For this you have to unwarp the physical tracked texture template and apply this to the material of the 3D model.

Hints

- Use the *CornerProjection.cs file* (found on BB) as a starting point for solving this task.
- You are allowed to use OpenCV's `Calib3d.findHomography` and `Imgproc.warpPerspective`.
- To avoid problems, you can set your unwarped images to use the same resolution as the texture files. (Note that the *fish* texture is not square - we recommend to start with one of the square templates.) You can downscale the texture files if you experience performance issues.

Explain: In your project description describe the theory for how the homography is estimated. You may use materials from the lecture slide(s) for your explanation. Please include figures/sketches and matrices.

b) When the texture template is tracked, display the default texture over it, so that you can see it in the video feed. For this you need to perspectively warp the image corners of the texture onto the 4 corners of the image target by using the inverse of your homography. Do NOT simply add a quad with the texture to your image target. Demonstrate your solution by including the corresponding code snippets in your report, and explain how it works.

Min / Total = **5.0pt / 10.0pt**

2. Interaction

Tasks

a) Create a new Unity scene, choose at least one image target, and attach some 3D model to it (you're free to use any of the assets we have shared with your throughout the course, or find new assets online). Implement an interaction technique by combining TWO of the following features in a way that makes sense. This should allow you to interact with your virtual model(s), or make virtual models interact with each other. The interaction must be based primarily (but not exclusively) on tangibles (e.g., movement of image target, relations between image targets). Note that key strokes are only allowed for debugging purposes, switching scenes or modes etc., but NOT for triggering interactions that are part of your tasks.

Feel free to be creative and have fun!

- **Color picker feature**
  This involves developing an interaction technique for changing the color of a material. Allow selecting a color from a predefined palette with at least 5 different colors. You should enable to continuously adjust the brightness of the color, e.g., by interpolating between black and the selected color. You may NOT use any finished color picker from the Asset Store, or a Unity Library, etc. but must instead implement this yourselves. Include code snippets of how you created the color palette and how a particular color is selected.
- **Text entry feature**
  Text entry is challenging in AR - find a way to support text entry through tangibles (i.e., image targets). You can use Unity's 3D text functionality to display text, e.g., in the local coordinate system of an image target. Support at least 26 capital letters + whitespace, as well as an interaction technique for deleting characters (deleting one character at a time

from right to left). Remember that you may only use the keyboard for debugging. Again, please provide code snippets that show how you create the text input interface and detect selection/deletion of a letter.

- **Finger tracking feature**
  Using OpenCV, you can support finger tracking. Start by unwarping the *finger_drawing_template* image target. When the user places his finger on the round white spot, you can pick the color of the finger tip (see video in the aux-material). Then you can use this detected color to extract similar colors from the unwarped image. Find the fingertip of the second hand within the large white space of the image target and render a circle around it. You can then use the x and y position of the detected finger on the plane to either paint, manipulate an object's position, etc. **Hint:** For simplicity you can assume that the finger always points from below (i.e., you do not need to support different orientations of the hand).

Hints:
We encourage you to re-use concepts from your group assignments, e.g., for selecting color or text you could use the position of an image target in local coordinates of another image target and set values depending on said position.

Min / Total = **10.0pt / 20.0pt**

Formal requirements for project handin

To successfully complete your individual project, you must upload a ZIP archive to Blackboard before the deadline, including TWO files: 1. a video of your prototype (mp4) and 2. a written project description (pdf). *Please read the following specifications carefully.*

1. Prototype video:
Record a short video to showcase your homography solution and interactive AR prototype. It should clearly show the functionality you implemented. Note that your video editing skills will NOT be assessed - a simple screen recording/smartphone video is good enough.

Requirements:

- **Format:** .mp4
- **Duration:** max. 2 minutes
- **Title screen information:** name and student number
  (you may also add a project title, "AR 2021", ...). If you don't have video editing software, this can be handwritten on a piece of paper and filmed, or perhaps presented in AR by attaching it to an image target.

2. Prototype description:
Write a short description of your AR system. Make sure you address each of the following points. (A short line per point is enough, but please make sure to write complete and correct sentences.)

○ For each task, describe which coordinate systems are at play and how you calculate the right transformations.
○ Include code snippets to show the most interesting parts of your implementation.
○ Explain how the homography works in principle, and how you implemented your solution.
○ Explain the overall idea you pursued for interaction, i.e., how you combined the two features you chose to implement.
○ Provide details on the interactions your prototype supports (refer to lecture slides on Interaction Techniques).
○ Shortly describe an alternative way for supporting this interaction, e.g., if you could use other hardware (refer to lecture slides on Display and Tracking technologies).
○ State which concepts you applied from previous weeks (group assignments).

We recommend using mathematical notation where appropriate and include images or sketches.

Requirements:

- **Format:** .pdf file
- **Length:** 1-2 pages
- **Title page information:** name and student number
  (you may also add your project title, "AR 2021", ...)