TDT4195: Visual Computing Fundamentals

Project Fall 2016

October 19, 2016

Bart van Blokland and Aleksander Rognhaugen
Department of Computer and Information Science
Norwegian University of Science and Technology (NTNU)

General

- **Delivery deadline: November 22, 2016 by 22:00.**
- **This project counts towards 10 % of your final grade.**
- You can work on your own or in groups of up to three people.
- Deliver your code along with the slides of your presentation on *itslearning* before the deadline. The final slide of your presentation *must* include a description of each group member's contribution.
- Please upload your presentation as a PDF file, and package your code into an archive (e.g. zip, rar, tar).
- There will be an evaluation presentation of your system. You should showcase your working system and explain the techniques that were used to solve the problem. Are there any limitations? What was difficult and what was easy?
- Ensure your code is documented and as readable as possible.

Digital Image Processing

- You may use the programming language of your choice. However, it might be a good idea to select one that supports matrix and image processing operations, e.g. MATLAB and Python with NumPy. *The lab computers at IT-S 015 support Python and MATLAB.*
- You may use already existing library functions.

Computer Graphics

- Your code must be runnable on the lab computers at IT-S 015.
- All tasks must be completed using C++. Use the Gloom project as basis for the Graphics part. You are allowed to use code written during the Graphics Labs, Graphics lab handout code, and the libraries included with Gloom. Do not include additional libraries.
- When using OpenGL functions, only use functions present in revision 4.0 Core or higher. If possible, version 4.3 or higher is recommended.

1 Project Description

Your task is to create two programs. The first program should be able to segment, locate, and record the type of all coloured shapes in a 2D colour image (see Figure 3). An overview of all valid shapes can be seen in Figure 1. You can assume that the objects are recorded with a simple camera setup showing a top-down view of a coloured checkerboard.

Figure 1: Overview of all valid shapes on the red and blue checkerboard. Notice that a black border has been added around the white hexagon for clarity.

The second program should use the location and shape type information to create, visualize, and animate a 3D scene using OpenGL. You should be able to select shapes and move them around the 3D scene using keyboard bindings.

A more detailed list of the minimum requirements can be found in the next section.

2 Requirements

Ensure that your programs work for *at least* two of the four example images to get full score for each task.

Digital Image Processing [5 points]

Your program should be capable of the following:

- **[3 points]** Segment all occurences of all shapes shown in the input image.
- **[2 points]** Mark the centre position, or centroid, and shape type for each segmented checker piece. Detecting shape colour and orientation is not required.

You may assume that the objects do not overlap.

Computer Graphics [5 points]

Like the previous Computer Graphics assignments, it is mandatory to use OpenGL 4.0 Core or higher. Use the Gloom project utilised previously as your starting point. It is recommended to use `http://docs.gl/` to verify that you are solely using OpenGL functionality present in version 4.0 or higher.

- **[1 point]** Render a 3D scene containing the checkerboard shown in the background of the set of input images.

- **[2 points]** Using the locations and shape types detected in the image processing part, render each shape as a 3D (extruded) model in the location it was detected on the checkerboard. *Tip: Start by creating a list of 2D triangles of all shapes specified in Figure 1. You can then implement a function which extrudes each triangle individually, as is shown in Figure 2.*
- **[2 points]** Implement a means by which individual shapes present on the checkerboard can be selected[1], and moved around to different squares on the checkerboard. Transitions from one square to another must be smooth (time-based animation).

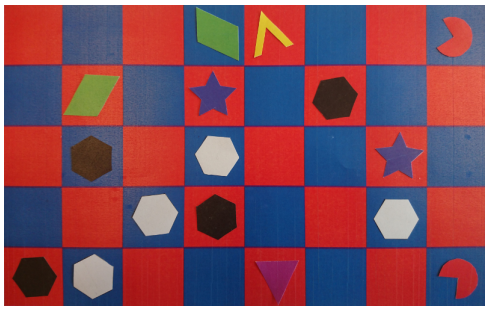
(a) A regular triangle.

(b) An extruded triangle.

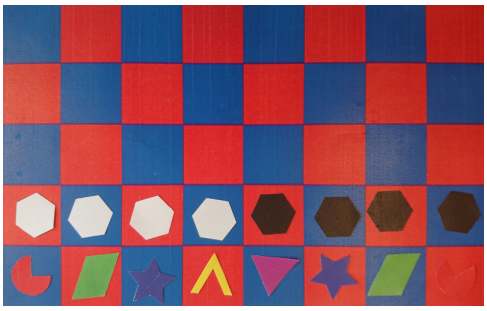
Figure 2: An example of an extruded triangle. (a) shows a single two-dimensional triangle, which has been extruded in (b).

2.1 Example Images

The example images in Figure 3 and more can be downloaded from *itslearning*.



(a) One image in which the various shapes are relatively distinct from the background.



(b) Notice the red Pac-man shape in the bottom-right corner of the image.

Figure 3: Example images with varying difficulty can be found on *itslearning*.

[1] One way of doing this is to bind a key that selects objects in a round-robin style.