

COMS30020 - Computer Graphics

Week 2 Briefing

Dr Simon Lock

Where are we ?

Hope that you all got RedNoise template working ?
If not, be sure to do so before next week !

Aim of this session is to introduce NEXT workbook
The practical is on Tuesday, so we need to be ready !

In this workbook we will actually do some drawing !
(but only very basic stuff - in 2D)

Later we'll be doing more sophisticated stuff in 3D
But we must learn to walk before we can run

Workbook Tasks

Interpolation a key technique in Computer Graphics
We will need to apply it to a wide range of entities:

Lines, Colours, Positions, Textures, Shading, Shadows

We first start with simple floating point values:

1.1 ? ? ? 5.5

We then move on to 3-element numerical values:

[1,5,3] [?,?,?] [?,?,?] [?,?,?] [?,?,?] [5,1,7]

So we can interpolate RGB colours, 3D positions etc

Final Objective

The tasks in the workbook will lead you step-by-step

Building up from simple tasks, to more complex

We'll provide hints, tips and advice along the way

Final objective is to produce the render shown below



What's your Vector Victor ?

This week we'll encounter a slightly annoying issue
We will be using 2 different things called "vector":

- "vector" (from std) used as a dynamic list structure
- "vec3" (from glm) used to store 3 element entities

Need to be careful - think about what you are doing
Ensure you use right structure for a given situation

In reality these are both inappropriate names
We don't get to choose names, so what can you do !

Prevalence of AI tools

Let's spend a bit of time talking about the use of AI
AI tools to support development are everywhere:

- Plugins embedded within IDEs
- Services integrated into code repositories
- "Freely" accessible web-based tools
(although many require a paid subscription !)

The lure to use AI in development is hard to resist !

Acceptance of AI tools

Rather than trying to ban the use of AI on this unit
(which would probably be impossible to enforce)

A more pragmatic approach is to permit such tools

In many ways, use of AI support tools is a key skill
Chances are you'll be using them when you graduate

Note that this isn't a "learning how to code" unit
So **careful** use of AI shouldn't conflict with ILOs

"Generative" AI

There is a reason why I put "Generative" in quotes
It's not really generating new material, but rather...
just recycling/reusing/recombining existing work
More like a "mashup" than creating something new

Just consider recent upset of Artists and Musicians
...and publishing companies:

<https://www.bbc.co.uk/news/articles/ckrrr8yelzvo>

"Write me an OXO game in Java"



ChatGPT



Sure! Here's an example of an OXO game (Tic-Tac-Toe) implemented in Java:

java



```
import java.util.Scanner;

public class OXOGame {
    private char[][] board;
    private char currentPlayer;

    public OXOGame() {
        board = new char[3][3];
        currentPlayer = 'X';
        initializeBoard();
    }

    private void initializeBoard() {
        for (int i = 0; i < 3; i++) {
            for (int j = 0; j < 3; j++) {
```

Mashup Coursework Submissions

The notion of a "mashup" codebase is nothing new
It does not rely on availability of effective AI tools
Students have been creating mashup work for years
borrow a bit of code from here, borrow a bit from there

Modern AI tools just make process quicker & easier
Students often don't even realise they are doing it !
But that's what you get when you Gen AI some code

Problem

How can we deal fairly with the following situations:

- Students who hand-craft ALL code themselves
- Students who "borrow" online code fragments
- Students who generate whole functions using AI

Typically people borrow/use AI when they are stuck

`I couldn't get my parser to work, so copied some code`

How can we let people do this (in order to progress)

But differentiate them from students who "did it all"

Concept Solution

If we could detect code that has been "derived"
We could factor it out during the marking process
(you don't get marks for code you didn't write)

This would allow us to give credit where credit's due
But allow people to fairly use AI when they need to

This would provide flexibility in development process
As well as encouraging people to think critically...
about material produced using "generative" AI tools

Implementing this Approach

We have a library of 300 solutions to the assignment
ALL submissions of ALL students from previous years

Many are available online (as part of code portfolios)
These will be part of the training set used by AI tools

These 300 solutions also borrow from online examples
(which provides us with an even broader sampling)

All we do is compare YOUR code with this training set

The more similarities that are found...

The more "derivative" your submission has become

I know what you are thinking...

There is BOUND to be some level of similarity
Some standard boilerplate code that everyone has
Or some convergent algorithm that everyone uses

You'd be surprised how much variability there is !
Also, we will use a variety of filters and thresholds
Aim being to factor out "natural" commonality

Also human-in-the-loop to filter out false-positives

The Bottom Line

You **may** use AI tools to support your development
However, avoid extensive use of AI code "generation"

AI tools should be there to ASSIST you in development
NOT to do ALL the work for you !

Be critical/analytical/reflective about the output of AI
It is not always correct, appropriate and useful

REMEMBER

You will only be rewarded for code that YOU write
Any "derived" code ("found" items or IA generated)
Will be discounted during the marking process
(resulting in a reduced final mark)

Why are we telling you this NOW ?
(rather than at the start of coursework period)

You will need the base code you write in weeks 1-8
As a foundation to complete coursework assignment

So the "derivation" rules apply for the entire unit !

Effective Use of AI

AI tools can often provide really good prompts
Integrated code completion tools can use helpful
(avoid following advice blindly - try to understand !)

Also useful for understanding errors/fixing problems
Paste obscure compiler error into chat for insight
Ask AI what is wrong with specific lines of code

Avoid dropping in large fragments of code...
Remember that AIs are always learning !

"Generative" AI Tools

"Generative" AI is good at performing small tasks
e.g. creating a first-cut attempt at a specific function
Clearly defined jobs, expressed in a short query:

write me a c++ function to parse an OBJ file

They're not well suited to creating complex systems
Code that has multiple interwoven features
Where expressing exactly what you want is hard

Effectiveness depends on "asking the right question"

Chat-Driven Software Engineering ?

It may seem like chat UI is ideal for working with AI Suits some tasks, but not Software Engineering !

Chat AIs require you to keep conversation going
Hard to manage a large codebase in a chat interface

Must keep AI up-to-date with current version of code
(e.g. if you have manually added some functionality)

Big challenges for long-term maintenance/evolution

"extend OXO to support more than 2 players"

Extending the OXO game code to include an arbitrary number of players becomes complex due to the way winning conditions are typically defined in Tic Tac Toe.

Here's why:

- **Winning Conditions:** Traditionally, Tic Tac Toe is played with two players ('X' and 'O') who take turns marking the board. The goal is to get a line (row, column, or diagonal) of your symbol.

With more players, defining winning conditions becomes ambiguous. Should a player win by getting a line of their symbol, or should it be the last player to mark a valid cell on a full board?

The AI says "no"

"rewrite OXO so that it doesn't use arrays"



ChatGPT



Certainly! Here's an example of an OXO game (Tic-Tac-Toe) implemented in Java without using arrays:

java



```
import java.util.Scanner;

public class OXOGameWithoutArrays {
    private static final int BOARD_SIZE = 3;
    private static char[][] board = new char[BOARD_SIZE * BOARD_SIZE];
    private static char currentPlayer = 'X';
```

Knowledge of codebase is key

You can't entrust the entire development to the AI
YOU need to understand code in order to evolve it:

- Where is the best place to add a new feature ?
- How do you refactor when things get messy ?
- How do you ensure full test-case coverage ?
- How do you fix something you don't understand ?

Ever tried to fix or extended someone else's code ?

How easy is it to evolve a black box of AI spaghetti ?

"AI tools will NOT replace YOU in your job"
but...

"Developers who use AI (well) are definitely going to"

- Pramod Dutta

Be careful out there !