# CS 330 Module Four Assignment Video Transcript

## Camera and Input Devices

Welcome to our weekly presentation. In this week we are going to look at the camera that controls what we see, and the input. Input, meaning we want to control the camera, and make it look at what we want. And the two inputs we'll use will be the keyboard and the mouse. By the end of this tutorial, you're going to learn how to navigate through your world by using the mouse and the keyboard.

So first, we're going to look at an example from LearnOpenGL that shows you a rotation. So it'll show you a few boxes and then the camera will rotate around these boxes. And then, we are going to modify it so that we can control that rotation. So once we understand how it rotates we're going to control it, so we make it go right or left rotate. Move the camera right or left based on our control, which is the keyboard. So if we press right it's going to go to the right, press left to go left, or up and down which is forward and backward, whatever.

And then, we're going to look at how to use the mouse to control that camera, more on how to use these two input devices to control the camera. Then we're going to look at bringing it all together. We're going to look at other input devices. And finally we're going to summarize all the skills gained in the video.

So, autorotation. Let's look at the current code. This is the code from LearnOpenGL. If we run it, you'll see this world. It hasn't of course — we didn't make any modifications to it or anything, so it is as-is. And if you notice, these objects are really rotating around the center point here. And this cube, since it's on that center, it rotates around itself. But the camera is really looking at that point all the time as it moves.

OK. So let's look at the code. We already learned everything we need — well, except I know we didn't cover the shaders or texture yet. So other than texture, which is next week, we learned everything about learning cubes and placing them and such. So we're not going to go over that. We're going to look at one element really, the command that controls where that camera looks at. And we almost looked at it in a previous presentation where we looked at 3D objects, a little bit but we didn't focus on it. It's actually this command.

This changes the view to force the camera to look at a certain point. It takes three parameters at the point to look at the direction up and down as such. Please review last week's view matrix to know what these parameters are, and that's it. All you have to do is just modify the view and then the camera will adjust itself accordingly and start moving. If you notice that the x and y are multi — are — first of all, we are inside the loop. Of course, I should have mentioned that. You should know that by now, everything we do is inside the loop. So it's continuously being checked or recalculated.

We recalculate the camera x and y based on a formula. Reposition the camera, or make it look at that new location. So we adjust our view matrix accordingly, and then that's it. And we redraw the shapes. And it will place them in that — well usually it's the same location, it's just the camera is looking at it from a different angle. And this is the formula. Now, let's make it a bit clearer. For example, because the values change in the loop, you keep looking at the camera rotating. Well, what if we hardcode this? So if we just say 0, 0. We don't want this rotation to change. So let's run it.

So now we are inside the loop and we are looking at 0, 0, and obviously, 0, 0 is looking somewhere else. You think you lost access, or you lost the objects or whatever, no. They are being drawn, but you're looking at the wrong location. Let's try to move the camera back. Let's say, minus 5, a little bit back. And let's run it. There you go. So the objects were drawn almost behind this, a little bit behind this when we were focused on the 0, 0. Let's move it back even a little bit more, let's say 15. There you go.

So these are the objects, and notice there is no more rotation because we just hardcoded the way the camera looks — or, the point the camera looks at, and we're done. It's looking at — from that position, minus 5, 0, is looking at the center. Fine, it's looking at here 0, 0, 0. OK.

Now, just as a small note, previously it rotated because it recalculated the sine and the cosine, the angle. And it changed because this is linked to time, while this is not a constant. Radius is, but this is time, so it gets time and time is different at each loop. And then it gets another. I think it's a double and then multiply with the radius.

OK. So now we understood that hey, this is fixed, the camera is not rotating, good. Now, how do we control it using the keyboard? So now we're going to look at controlled movement. So let's see. We want to be able to read the keyboard. Now we already have something built in, but we never looked at it because it wasn't the time. Now is the time. There is a function here called “processInput” that we have here and it's inside the loop. Anything that happens to the window immediately calls this function, processInput. This is the function that checks if something happened to the window.

Well, let's see, what does it do? Right now it keeps checking, “Hey, did he press Escape?” This is the function “GLFW\_GET\_KEY.” And the parameter is on this window, “Did he press the Escape button?” See, “Was it pressed?” That's what this means. If yes, it calls a function that closes the window. Very simple. So let's run it. Let's look. Here is the window. I'm going to press Escape, and there you go. The OpenGL window closed, notice.

OK. So that's it. This is how we control the keyboard, or how we read input from the keyboard. Well, let's modify this and say, else if. And let's say — I don't like typing, so I'm just going to copy/paste that part. Now I want to say, “Hey did he press, let's say, up and down?” Up means move forward, the up key, and back is down, move backward. So, “Did he press up?” So “GLFW\_KEY\_UP.”

Notice, you can choose any key you want, right, up, down, left, even the letters, F1, F2, whatever you want. I just want up. If it was pressed…here we go. I want to do something. Now notice, let's go back to our small example in here. When I wanted to move the camera backward or forward, I didn't move it forward, I moved it backward. I changed the z, which is camZ, which is the variable used in here, interesting. But this variable is local. I mean, logically, you're going to say, “Hey it's simple.” Let's say, “camZ minus equal all point,” let's say, 1, that's correct. You are changing this variable, but this variable is local. Well, local here. It's declared here. That's easy. Just take it out. I usually don't like to declare variables inside the loop. So I'll take all three, even though I don't need all three to be global, and we'll put it here. And that's it.

Notice that now this is declared. Notice that it's moving the camera back by this much. You could change this value only if you press the key “up.” Well OK, let's run it. Up, up, up, up, notice, up means I'm moving backward. Even though that's wrong actually, so really we need to move forward if you press up. So let's change this to plus, and let's hit two birds with one stone and fix this — not fix it. Adjust it, say down, if he presses down then move back.

So now we have two keys, up and down, and you can modify it, right and left. OK, and you can go through our reading material to see oh, what if I pressed Alt, and I want to do alt + something, you can do that as well. Let's see. So if I want to move back, I'm moving the camera back. I'm moving the camera forward, backward, forward, see? And guess what? You can modify the code now to move right and left. All you need to change is the camX. The variable camX which controls the motion on the x-axis.

If you want to move it up and down, not forward and backward, you would change the y variable. Do we have a camY? Let me see. Yeah we did it, I think he placed it at 0. Let's see the code. Yeah, you would assign a camY in here. And that would go up and down and so on. See, that’s how it is. That's how simple it is. You just read the value, determine which key got pressed, and modify it. Now here if you saw that the objects move way too fast, there’s a way to calculate sensitivity. By simply changing the increment value, you can control the sensitivity and the camera flow in your world.

Now, we want to see or learn how to read the mouse movement or clicks. So we finished reading — looking at the input from a keyboard, now we are going to look at the mouse. First, I declared a function. It's called “mouse\_callback” and it says, “This is the function that you need to call when a mouse is clicked.” Now it doesn't say that yet. This is just the prototype.

In main — not inside the loop, be careful, outside. This is like at the beginning when we create the window and open it and initialize it, we call the function and tell it that we're going to be using our function mouse\_callback. So attach this function to this window. So that's all that this does. It's a callback. If something happens to that window, call this function. And in here we issue a command telling them, OpenGL, “Hey, using this window, which is our window, it's our variable right up there, you can see it, read mouse input. Anything that happens to the mouse, read that input and direct it to the window.”

And guess who's going to capture it? This function. OK. And that's it, we don't put anything in the loop — the while loop — like we did with the keyboard. This is an event triggered function where the minute you move the mouse it'll keep listening — Is the mouse moving? Is the mouse moving? — by itself.

And then of course we declare the function, and here it is. So it's a function. Three parameters, x, y, and z — the window, sorry, and then x and y. And in here I just used a simple function called getMouseButton. This is an OpenGL GLFW function that reads the button, or it reads the mouse, or something got clicked, moved, dragged. And you specify, what do I look for? “Hey, did he press the left one — the left button?” If this function returns true, you'll notice I just made the code like this. Initialize back the camera to where it was before we started running the code.

So this logic says, “Hey, if he pressed the left button, put back the camera at the original location.” Remember, we changed it to negative 10 so you were able to see the objects, right at the beginning of this video? Well, when you moved with the keyboard front and backward, then you changed this variable. So you changed this variable, camZ. Well, when I press the mouse, I put it back to — I reinitialize it to negative 10. That's what it does. Let's run it.

So I'm moving forward. I'm moving backwards, forward. Now I'm going to press the button. See, the minute I clicked the button, in it, everything got initialized because we just reinitialized the variables. And in the loop it just says, “Hey, using these variables, move the camera there.” Here, this command. Immediately when I press the button here I can go back, back, back back. Now when I press the left button, there you go.

Now, first I pressed it several times because the mouse was outside the window. The mouse has to be inside the window because we disabled it. You cannot see it. So you could not disable it if you want to be able to see the mouse. Usually we don't. You don't want to show the cursor, the mouse cursor. Here, we disabled the cursor. So if you click and nothing happens that's because your mouse is not within the window frame. The mouse must be inside the window for the window to read your event.

Now, let's bring it all together. You learned how to work with the mouse and the keyboard, but there are a lot of other devices, such as the virtual reality glasses, the sensor glove, input devices, audio-activated devices. All these input devices can be used to read input. And each one has its own libraries that you can explore to link to and — to link your code to it. But remember, each one has its own library, and its own functionality that needs in-depth exploration.

So, skills gained. Right now, you have the control, or the power to actually control the camera. Move it any way you want. You can explore the different formulas out there to rotate it. Well actually, there aren't that many formulas, really. You can now rotate it on x, y, and z; x and y; z and x; whatever you want. All you have to do is just control the three variables: in our example, camZ, camX, camY. We don't have camY, but you can add it. And that just controls it. And you can read inputs from the mouse, from the keyboard. Remember, when you read an input from the keyboard you can do other things, like play music or play sound or change the texture, which later you'll learn how to do, and that's it. I hope this helps and good luck.