AME 394

Compositional and Computational Principles for Media Arts

Instructor of Record | Matthew Ragan

Description

In much of today's contemporary media practice there is a tight coupling of compositional form, content, and underlying computational mechanisms. This integration holds the potential to yield new modes of expression and wholly new art experiences as is evident in emerging forms of real time generative art, network-based art, game-based art, and interactive performance. As both practitioners and participants, we must develop a critical understanding of the relevant compositional and computational principles that frame this work. In this course, students will develop a working understanding of fundamental compositional and computational principles, and apply their understanding through the realization of exploratory media artworks.
Student Learning Outcomes: Upon successful completion of this course, students will be able to:

- identify opportunities to use modular based programing methods in your own work.
- experiment with dynamic media structures.
- compose interactive TouchDesigner networks that use sound and video.
- communicate with other programs or computers over a network.

Course Software/Websites

The course, and its assignments, will use Derivative's TouchDesigner.
Course readings and example code will be posted on blackboard under Assignments.

Course Requirements

It is expected that all students will be prepared for and actively participate in all class sessions. This means, reading assigned texts, viewing any assigned video completing class assignments, asking questions, answering questions, providing relevant commentary and participating in group activities when necessary. Please approach each class with an open mind and your full attention. If a significant portion of the class is not coming prepared I reserve the right to make additional assignments to ensure the reading is being done.

It is the student's responsibility to be aware of points earned and their standing in the class.

Please feel free to ask the instructor at any time for suggestions on how to improve or about your current status in the class.

It is each student's responsibility to complete assignments even if they are absent from class when they are assigned or due. Late assignments will not be accepted unless previous arrangements have been made with me.

Most of the course assignments include writing programming code. Just like any other assignment, copying someone else's code or sharing yours with someone can be considered cheating. However, helping people either during in-class work times or on discussion forums is encouraged - just do not do other students' homework.

It is necessary for any media you use to indicate where you acquired it. If self-made, please indicate. If you found it online or somewhere else you must indicate the source.

Course Content Outline

Dates	Day	Reading / Assignments	Concept / Unit
8/25	*Mon*		Class overview, policies, and materials
8/27	*Wed*	First Things to Know about TD Artist or Installation Share	Welcome to TouchDesigner - Learning the Interface
9/1	*Mon*	No Class - Labor Day	
9/3	*Wed*	Bring 10 Images / Videos to Class Look Through Op Snippets	**Principles of Design**
9/8	*Mon*	Continue Principles of Design Look through Op Snippets	Composition
9/10	*Wed*	me.time Snippet work time **Snippet 1 - Exploration (Due Friday)**	Composition
9/15	*Mon*	Assignment Review Referencing Review Efficiency - How do I know? Working with Noise Project 1 Explanation Your first Button	Composition
9/17	*Wed*	Your First 3D Scene Project 1 work time **Project 1 - Composition (Due Friday)**	Composition

9/22	*Mon*	Rushkof Program or Be Programmed Simple Sound Example	**Making Meaning**
9/24	*Wed*	***Why CHOPs?*** Constant, Speed, Count, Filter, Lag, Noise, Timer, Merge, Cross, Switch, Mouse In, Panel, Keyboard	Manipulation
9/29	*Mon*	What are Panels Your first Script Panel Execute DAT	Manipulation
10/1	*Wed*	**Snippet 2 - CHOPs (Due Friday)**	Manipulation
10/6	*Mon*		Manipulation
10/8	*Wed*	**Project 2 (Due Friday)**	**The Interface**
10/13	*Mon*	No Class - Fall Break	
10/15	*Wed*		Control
10/20	*Mon*	**Snippet 3**	Control
10/22	*Wed*		Control
10/27	*Mon*	**Project 3**	Control
10/29	*Wed*		**Embodyment**
11/3	*Mon*		Interaction
11/5	*Wed*	**Snippet 4**	Interaction
11/10	*Mon*		Interaction
11/12	*Wed*	**Project 4**	Interaction
11/17	*Mon*		**Bigger Faster Stronger**
11/19	*Wed*		Communication
11/24	*Mon*	**Network Analysis**	Communication
11/26	*Wed*		Communication
12/1	*Mon*		Communication
12/3	*Wed*	In Class Work-Session	**Wrap-Up**
12/5	*Mon*	**FINAL - Final Presentations**	9:50-11:40

Assignment Descriptions

Snippets

Op Snippets 1 - Exploring and Getting your Feet Wet

Operator Snippets are one of the most useful help features in TouchDesigner. By the end of the semester we'll be submitting our own snippets to derivative, before we're ready to do that, however, we first need to get a sense of what is already there.

For this assignment, you'll spend some time exploring the TOP Operator Snippets. You'll identify three operators that you find interesting, and then create some simple examples that demonstrate what those operators do. Start by looking through all of the TOP examples and find three TOP Operator Snippets that you like / find useful / find interesting.

1. Create three Containers, each focused on a different operator.
2. Inside of each container create a network that demonstrates what a given operator is used for. Each container should contain:
    a. At least 3 operators
    b. No more than 10 operators
3. Containers should have the same dimensions as the final TOP in your network.
4. Every Container in your network should be commented.
5. Every example should be animated in some way - using expressions or global variables (me.time.seconds / me.time.absSeconds or me.time.frame / me.time.absFrame)

Op Snippets 2 - Building a Nervous System

We've spent some time already talking about the importance of modular design when it comes to programming, and this assignment is an opportunity to try your hand at building a modular CHOP control network.

For this assignment, you'll spend some time exploring the CHOP Operator Snippets. To help you get started, I've built a component for you to animate. Build a CHOP network based on what you learn in Op Snippets that controls the ten different parameters that are already assigned in the tox file. Pay close attention to the readMe DAT in the provided TOX file, as how you name your channels is very important.

1. Your snippet should use a Base to encapsulate your CHOP network.
2. Your snippet should Include the 10 correctly named channels to drive the provided component.

3. All 10 channels should be animated in some fashion – in other words, there should be some change (no matter how small) in the channels that drive the connected component.
4. Your network should be well commented – more than 1 readMe DAT. Your comments don't need to be especially long but should help me understand what you were thinking / working towards. Include your name and a date stamp in each comment.
5. Your network must contain each of the following CHOPS: Merge, Rename, Speed, and Math. As a note, you can use more than one of these, but I want to see at least one of each of these in your network.

Op Snippets 3 - Buttons and Sliders Everywhere

The interface is the intersection space between your code and the user. Users rarely touch the code itself in an installation or performance. In fact, programmers hardly touch the code themselves once a tool has been built. Instead programmers and users alike pass through an intermediary, the Interface. Interfaces are so common to us, that we often forget they exist. We forget that they impose invisible limitations on how we can manipulate the digital world. We forget that they are the part of a program that we interact with most often, and have the deepest relationships with.

For this assignment, you'll be practicing interface building. Specifically, you'll be making your own reusable elements that you'll implement in the next project. I'll want to see that you can build a Button with a name (or colors) that change according to its activation state, a Horizontal Slider, a Vertical Slider, a 2D Slider, and a Float-Feedback Slider. It's important that you make each one of these control elements, as I'll want to see at least one of each in your next project. You'll turn in a single .toe file with each of the following:

1. A Button that:
    - has at least two different text values whose display varies dependent on the state of the button.
2. A Horizontal Slider that:
    - has a label (DAT to Text TOP Method) for the background (sized to match the dimensions of the parent)
    - is correctly scaled so as not to fall off the outside bounds of the container.
3. A Vertical Slider that:
    - has a label (DAT to Text TOP Method) for the background (sized to match the dimensions of the parent)
    - is correctly scaled so as not to fall off the outside bounds of the container.
4. A 2D Slider that:
    - has a label (DAT to Text TOP Method) for the background (sized to match the dimensions of the parent)

    - contains a square knob and cross hairs.
    - contains an Out CHOP with two channels labeled: x and y
5. A Float Feedback Slider that:
    - has a horizontal slider with a label, and is scaled so as not to fall outside of the bounds of the container.
    - the slider should contain an In DAT, a DAT to CHOP, an Override CHOP, and a CHOP Execute DAT.
    - has a field COMP

Projects

Project 1 - Composition

Beyond just the mechanics of programming, we also need to consider the weight and heft of ideas when we are composing images with code. When considering the composition of an image, where different elements live in the raster, using the principles of design can both give us a language for expression and a source of inspiration.

Drawing from our discussion about Principles of Design, this assignment is an opportunity to explore a single principle creatively. You can use the .toe file on blackboard as inspiration and reference for this project.

1. Select one of the following principles of design, and include a comment in your network about what principle you've selected (also include the text from the handout in BlackBoard):
    - balance
    - emphasis
    - movement
    - pattern
    - repetition
    - proportion
    - rhythm
    - variety
    - unity
2. Your project should then be an exploration of the selected principle, and an expression of the ideas embedded in it – this should be communicated visually in your final composition.
3. Your Network must include an image that you used as a part of your research for this project.
4. Your project should be encapsulated inside of a container Component that has a resolution of 1024 x 768.

5. The resolution of your Container should be set with an expression – you can drive your resolution from the contents of a table, or the resolution of another operator – for example op('filePathToAnotherOperator').width
6. Your network must be well commented. Include your name and the date in any of your comments.
7. Your network must include at least 10 operators.
8. Your final composition should be animated.
9. Your network should include a button to toggle between your inspiration / reference image and your final composition
10. Your final network should run at at least 40 frames per second.

Project 2 - Manipulation

At this point we've created a composition in a container (Snippet 1, Project 1) and we've built a control system of animated channels (Snippet 2). Now we're going to start to look at how we put those ideas together.

Project 2 builds off of what we've learned in our other projects, and act as an extension of Snippet 2. For this project you'll use what you've made in Snippet 2 as a starting point for building your own modular set of visual and animation elements. Using Snippet 2 as a guide, you will build two distinct components:

- Your own visual component (Container) that has 15 control elements (like the tox file that I supplied to you).
- A component (Container) that holds a CHOP network which animates your visual component. This CHOP network should output 15 control elements.

1. Your network should contain a Visuals COMP (Container) – make sure this is labeled with your initials and "vis". For example, mine would be labled, "mrVis".
2. Your network should contain a Control COMP (Container) – make sure this is labeled with your initials and "CTRL". For example, mine would be labled, "mrCTRL".
3. Your Control COMP should contain 15 animated channels.
4. Your Visuals COMP should contain a constant with 15 corresponding "default" channels that match the names in your Control COMP.
5. The Visuals container should use expressions or CHOP exports to connect the 15 control elements in the network from a single Null CHOP. All 15 channels should make a recognizable change in your final image.
6. Your Visuals COMP should contain a text DAT with a script for setting the resolution of the artwork in your network using storage.

7. Your Visuals network should contain a real-time rendered piece of geometry – this requires a render set up with a Camera, Light, and Geometry COMP, as well as a render TOP with a resolution of 1280 x 720.
8. The resolution of your Visual COMP should be 1280 x 720 – using storage as a method for recalling those values.
9. Both your Control Container COMP and your Visuals Container COMP should be well commented.
10. Your final visuals COMP should perform at least 40 fps.

Project 3 - Control

At this point we've created a composition in a container (Project 1, Project 2), we've built a control system of animated channels in a container and linked those parameters to our a visuals system (Snippet 2, Project 2), and we've built our own custom control interface elements (Snippet 3). Now we need to think about how we build an interface to control and operate our visuals without needing to be in the programming environment. What does a user / operator get to drive? In what range? What are the limits that we impose on the user / operator? What message does this communicate to your operator / user?

Project 3 builds off of what we've learned in our other projects, and acts as an extension of Project 2. For this project you'll use what you've made in Project 2 as a starting point and build a control interface for your set of control CHOPs. We'll use the control widgets that we built in Snippet 3 for this, looking closely at how we build a control panel made up of sliders and buttons. The project you submit will be a single toe file that contains your visuals network and your control panel. Your final project should have the following:

1. A well composed and organized control panel.
    - consistent spacing
    - use of me.digits for layout organization
    - easy to read / understand labels
2. A total of 15 control elements:
    - You must use at least one of each of the following:
        - button
        - horizontal slider
        - vertical slider
        - 2D slider
        - float feedback slider
3. At least one radio button group.
4. At least one use of hierarchy as an organization method.
5. Buttons and sliders that have descriptive labels.
6. Your visuals COMP must contain a single button that opens your control panel as a floating window using a script. This button should only be visible when moused over. Example code:

    - ctrl = op('../../ctrl')
    - ctrl.openViewer()
7. The size of your control panel should be 1024 x 768. Use storage to set and retrieve these values for your Control COMP. Hints:
    - Store values with the script – me.parent().store('key' , value)
    - Fetch values from storage with the expression – me.fetch('key')
8. Both your Control Container COMP and your Visuals Container COMP should be well commented.
9. Your final toe file should be set to open in perform mode.
10. Your final visuals COMP should perform at least 40 fps.

Final Proposal

Here we are, quickly sprinting towards the end of the semester – which means it's time for us to start thinking about our final projects. We've now covered the basics of building an application from beginning to end in TouchDesigner. As a quick re-cap, here's what we've learned to do so for in class:

- **how to work with images or movie files** and modify / change those files with various operations
- **signal flow** – passing video, channel, surface, or data information through a network of operators
- **encapsulation** – how to compartmentalize our networks into smaller pieces or components (our visuals network and our control network)
- **referencing** – how to call for the parameters of other operators in order to make changes in a network
- **rendering** – how to render a piece of geometry in real time (including how to light, position, and view that geometry)
- **scripting** – creating feedback, and opening windows with scripts
- **interface component building** – building single reusable elements like sliders
- **interface building** – how to build whole interfaces
- **setting up applications to run in full screen from perform mode**
- **saving files to open in perform mode** and bypass the programming environment.

Whether you believe it or not, you're now all ready to build your own applications and tools, and you're going to use the next couple of class assignments to get there. This first assignment is the proposal stage. What is it that you want to build? What's it going to do? How are you going to control it? Does it connect to another project? Do you need to be able to send data to another programming environment? Do you need to be able to receive data from another programming environment? In short, I want to know what you

want to build, why, and how you plan for it to work. This is our planning phase, so I want to see your big plans. For this project you'll turn in the following:

1. A short narrative (250-500 words) description of the project. What does it do? What does it look like? Why is it important to you? What idea or concept are you exploring (what ideas or concepts interest you… if you feel stuck, remember you can always return to our first assignment and the principles of design)?
2. A system diagram – create a simple flow-chart that tells me what hardware and software is involved.
3. A list of the problems challenges that you know how to solve, and a list of the problems and challenges that you don't yet know how to solve.

Final Project Requirements

As much as I'd love to let you have free reign on the final project, I do want you to have a few guidelines in place. With that in mind, plan your final project knowing that I'll want to see you:

1. Create a visual composition with your assignment (you don't have to build a visualiser, but I do want to see that you're using what we've learned about working with pixels… if you're not sure if what you're thinking about counts, ask me)
2. Create a control panel for user interaction / programmer interaction.
3. Perform in real time, or close to it (I don't want to see projects that crash or hang).
4. Be organized and modular in your programming (I want to see that you know how to use encapsulation to make sure your projects are organized and reusable).
5. Comment your code – it will make you all the happier when you're building something over the course of several weeks.

Project 4 Final Project First Draft

The end of the semester is fast approaching, and I want to see where you are in the process of working on your final project. Remember that your final project needs to include the creation of some visual element, and the use of a control panel. For this project I'd like you to turn in the first draft of your control panel. This doesn't have to be complete and it certainly change – I just want to see how you're thinking about control. What elements are you driving from your control panel, are you using radio buttons toggles, sliders, 2D-sliders, how well organized is your control interface, etc.

For this project, you'll submit a toe file with the beginnings of your control panel. It should include:

1. A container COMP labeled control (really it can be labeled anything you want, I just want to see that you've given the control container a unique name).
2. Control Elements (sliders, buttons, fields, etc.)
3. Labels for your control elements, and headers for button groups (this might not apply to your project, but they should be there if it does).
4. Comments, comments, comments.
5. A readMe DAT – include a text DAT in the root directory of your network that outlines what you've already built in this draft, and what you have left to build for the final. Include questions, challenges, and discoveries you've made as you've started working on your final.

Final Project

To get us started, I want to make sure that we revisit the project requirements that we talked about with the final proposal process:

Final Project Requirements

As much as I'd love to let you have free reign on the final project, I do want you to have a few guidelines in place. With that in mind, plan your final project knowing that I'll want to see you:

- Create a visual composition with your assignment (you don't have to build a visualiser, but I do want to see that you're using what we've learned about working with pixels… if you're not sure if what you're thinking about counts, ask me)
- Create a control panel for user interaction / programmer interaction.
- Perform in real time, or close to it (I don't want to see projects that crash or hang).
- Be organized and modular in your programming (I want to see that you know how to use encapsulation to make sure your projects are organized and reusable).
- Comment your code – it will make you all the happier when you're building something over the course of several weeks.

With the above in mind, your final project, at it's heart, is an opportunity for you to practice the techniques that we've gone over this semester. Remember that you're building an application from scratch – it doesn't need to be perfect, but the toe file that you submit should include:

1. An originalProposal DAT – include a text DAT with the text from your original project proposal – you don't need to include the system diagram, just the text from your original project proposal.
2. Congruence with proposal – Your project should have some congruence with your original proposal. Your final doesn't need to be exactly what you proposed, but it should feel related to the idea you were originally considering exploring.
3. Performance – Your project should perform at at least 25 frames per second.
4. Organization – Your project should be well organized. We've spent a significant amount of time in class talking about organization and modular approach. Make sure that you're making use of encapsulation to create tidy networks that are easy to read.
5. Visual Component – Your project should include a visuals component.
6. Control Component – Your project should include a control component.
7. Comments – Your project should be well commented
8. Use of Storage – Your project should make use of storage in at least one container.
9. Perform Mode – Your final project should be configured to begin in perform mode.
10. A selfReflection DAT – include a text DAT in the root directory of your network that outlines your successes and lessons.
    - On a scale of 1-10 (1 – your project is totally different than you expected; 10 your project is exactly what you set out to make) how close did your final project resemble your originally proposal?
    - What challenges were you prepared to meet?
    - Where did you get lost along the way?
    - What are you especially proud of?
    - What were you especially frustrated by?
    - If you were going to do this project again, what would you do differently?

Friendly reminders

If you're submitting a project that uses photos, video, audio, or 3D object files you'll need to make sure that you zip those files together into a single folder.

Especially large project folders can be submitted by posting a link in your submission to a cloud storage platform – the 200MB folder is going to take a long time to upload to Blackboard, you can submit a link to a file on google drive, dropbox, etc. if you have a large project.

I might not have access to the unique equipment that you used for your project – if you're using a midi keyboard, an OSC interface, or another input device you'll need to have a switch in your network that toggles between live and static inputs (this is the same process that we used in project 2 when thinking about default values).

Late Work

Assignments will remain open after submission deadlines. Late work will be loose 2% per hour past the submission deadline. If you anticipate being unable to meet an assignment deadline, communicate with the instructor prior to deadline for the assignment in order to make other arrangements.