

This project was inspired by the [Game of Thrones intro sequence](#) and various [SimCity intro sequences](#). Your animation obviously does not have to be as detailed as these animations, but the basic idea is that you are going to be animating the construction of a city. You can be as creative and experimental as you would like with this but the general requirements are as follows:

- You must have at least 4 buildings, one of which must be made using a custom profile inside of the `cylinder()` function.
- All buildings not made using the cylinder function must animate into place (the buildings made using the cylinder function can also be animated, but that is not required).
- There must be a ground plane under all the buildings.
- There must be some sort of object animating in the sky above the city.
- At least one object in the scene that is not on the ground plane must rotate.
- At least one building must translate into place.
- At least two buildings must rotate into place, each along a different axis. For example, one building could rotate up about the x axis and another could rotate up about the y axis.
- At least one image colormap must be used.

Any additional effort made to further enhance the visuals or animation complexity of the final animation will be awarded bonus points. Some examples of things you can do for bonus points:

- Add a sky color (or better yet, a sky image)
- Animate more than the required number of buildings
- Include shadows
- Have a non-flat ground surface
- Animate multiple objects in the sky
- Have objects move in a more complex manner than simply translation or rotation (perhaps animate a truck through the city)
- Have the animation transition from day to night
- Play some music (using MATLAB) along with your animation
- Include multiple colormaps.
- Animate the view/camera (must be intentional movement; having the axis set to autoscale doesn't count)
- Have the animation react to keyboard or mouse input

Any other non-trivial effort beyond the requirements but not included in this list will also be awarded bonus points.

Meeting all of the requirements will result in a 100 for the project. Every feature above and beyond the requirements will be 4 bonus points each, up to 20 points, for a maximum score

of 120/100. No part of this assignment will be auto-graded; it will be graded by your TAs by looking at the animation your code produces.

To aid you in your animation endeavours, you have been given two helper functions. You can use the help command in MATLAB to view the full documentation for each one, but a brief description is given here. Again, you DO NOT have to write these functions, they are for you to use (or not use) however you wish.

`makeBuilding(xPoints, yPoints, zPoints)`

For all calls to `makeBuilding()` the first three inputs must be 1x8 double vectors representing the 8 corners of a building. The first input is the x coordinates, the second is the y coordinates and the third is the z coordinates. After these first 3 inputs, there are a few different things you can do.

`makeBuilding(xPoints, yPoints, zPoints, color)`

When a fourth input is specified, it is assumed to be either a single character describing a color or a 1x3 vector of an RGB triplet. The resulting building will be created using the specified color (if no color is specified, black is used).

`makeBuilding(xPoints, yPoints, zPoints, 'img', image)`

If the fourth input is the string 'img' then the function expects to find an NxMx3 image array in the fifth input. This image will be applied to all faces of the resulting building. Additionally, the fifth input can be a 1x6 cell array where each entry is an NxMx3 image array. In this case, each image will be applied to a single face in the following order: top, bottom, left, right, front, back.


`rotPts(points, anchor, degrees)`

`points` must be a 3xN vector where the first row is x points, the second row is y points and the third row is z points.

`anchor` must be a 1x3 vector of an [x, y, z] point defining the point in 3D space about which to rotate the points in the first input.

`degrees` input must also be a 1x3 vector specifying the number of degrees to rotate about each axis. The first element in the vector specifies the degrees to rotate about the x axis, the second input specifies the rotation about the y axis, and so on.

For more detailed information on how either of these two helper functions work, use the help command inside of MATLAB.

The default view when calling `makeBuilding()` is not very helpful for seeing what's going on. Instead, you can run the following snippet of code and use the rotate tool (  ) in the figure window to see how the function works.

```
makeBuilding([0 0 1 1 0 0 1 1], [0 1 1 0 0 1 1 0], [0 0 0 0 5 5 5 5] , 'r');  
axis equal  
view(3)
```

This code will produce a red building that has a 1x1 unit base and is 5 units tall.

Some additional built-in functions that may be useful to you are:

- `cla`
- `colormap`
- `patch`
- `surf`
- `mesh`
- `campos`
- `view`
- `drawnow`
- `pause`
- `meshgrid`
- `sphere`

Feel free to make your own helper functions as well if it helps you organize your code.

Lastly, we have provided a demo project, `demoBuildCity()` for you look look at to get a better idea of what is expected. The demo project does not need any inputs to run. The demo project meets all of the requirements and additionally has a background color. So the demo would receive a final score of 104/100. You do not have to base your animation around the demo in any way, as long as the requirements are met. To see what is possible with image colormaps, you can run the demo project with `'incImg'` as an input. This will map skyscraper walls to the sides of the buildings in the animation. With this enhancement, the demo score would now be 108/100.

When you get ready to submit your project, upload all files necessary to run your animation to T-Square. This includes the helper functions we provided, any helper functions you created, as well as any images or other files you use. Additionally, include a `README.txt` file briefly describing how to run you animation. For example, if your main function is called `superDuperCity()` and you need to input the rotation speed of the buildings into the function, then your `README.txt` file might look something like:

```
-----  
To run my project, type:  
superDuperCity(<rotationSpeed>)  
in the Command Window. <rotationSpeed> will specify how fast the building  
rotates in degrees per second.  
-----
```

## Homework 13 - Animation Project

---

You will demo your animation to your TAs the week of April 18th to 22nd in recitation. The only time your submission on T-Square will be looked at is to give you partial credit if you do not show up to your recitation to demo your project.

This assignment is much more open-ended than anything you have seen before in this class. You can name your functions whatever you want. You can make as many helper functions as you want. This is your chance to get creative without worrying about exactly matching a solution function or dealing with rounding errors or any of that. If you can dream it up, this is your chance to make it! Use all those crazy functions you've been Googling all semester! No go out there and have some fun building your city!