Task 1 :
To write an executable code that takes the input scene and the text prompt from the command line argument and outputs an image with a red mask on all pixels where the object (denoted in the text prompt) was present.
(e.g. python run.py --image ./example.jpg --class "chair" --output ./generated.png)

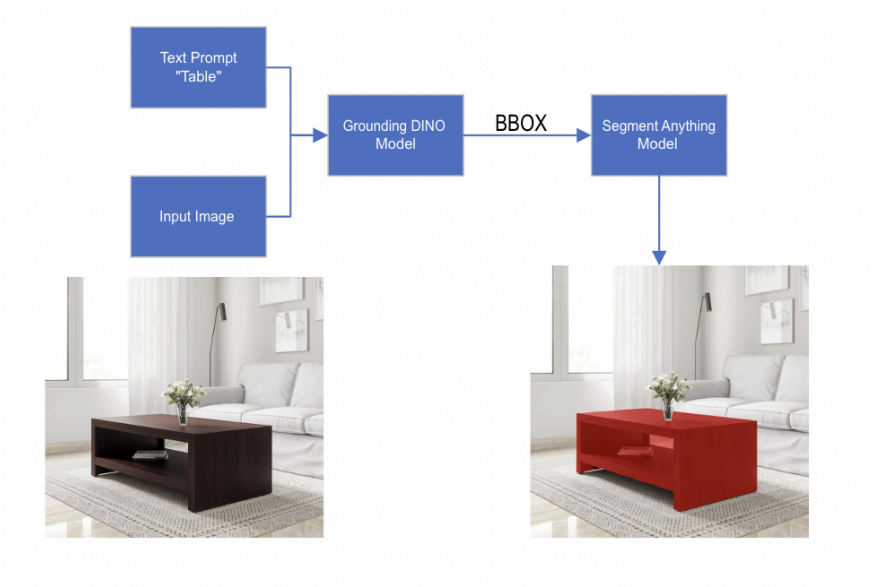
Approach :
Models Used :

Grounding DINO: To obtain bounding box coordinates of an object in an image given text prompt.

Segment Anything : To obtain the corresponding segmentation mask given bounding box coordinates.

Sample Input **Sample Output**

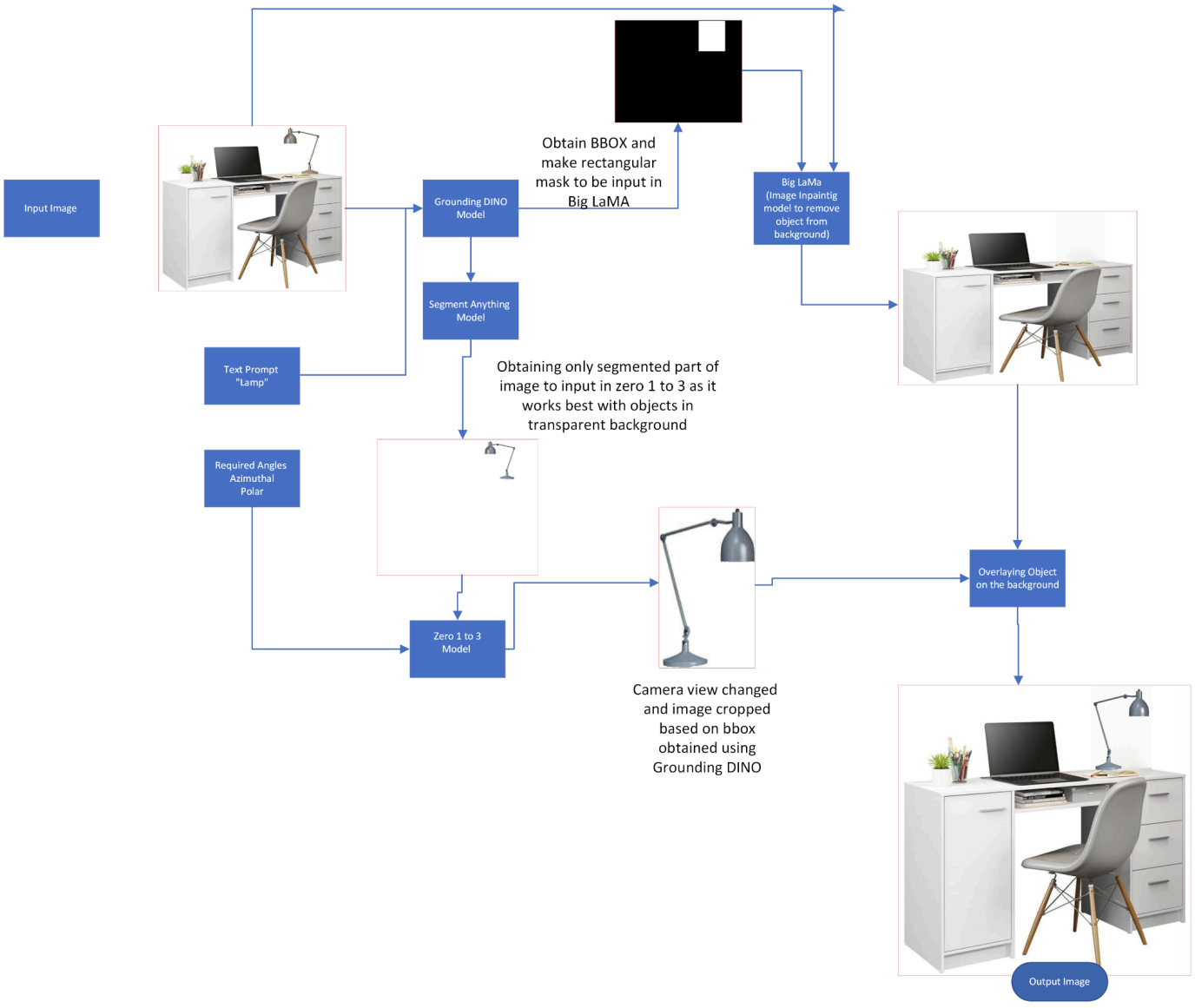

Workflow of Task 1



Task 2 :
The second task is to change the pose of the segmented object by the relative angles given by the user. You can use a consistent direction as positive azimuth and polar angle change and mention what you used.
(e.g. python run.py --image ./example.jpg --class "chair" --azimuth +72 --polar +0 --output ./generated.png)

The generated image:
Should preserve the scene (background)
Should adhere to the relative angles given by the user

Planned Workflow



Models Used :

Grounding DINO, Segment Anything, Big LaMa (for Background inpainting),

Zero 1 to 3 (tried hosted interface and also got it running locally but requires heavy computing to get any kind of output)

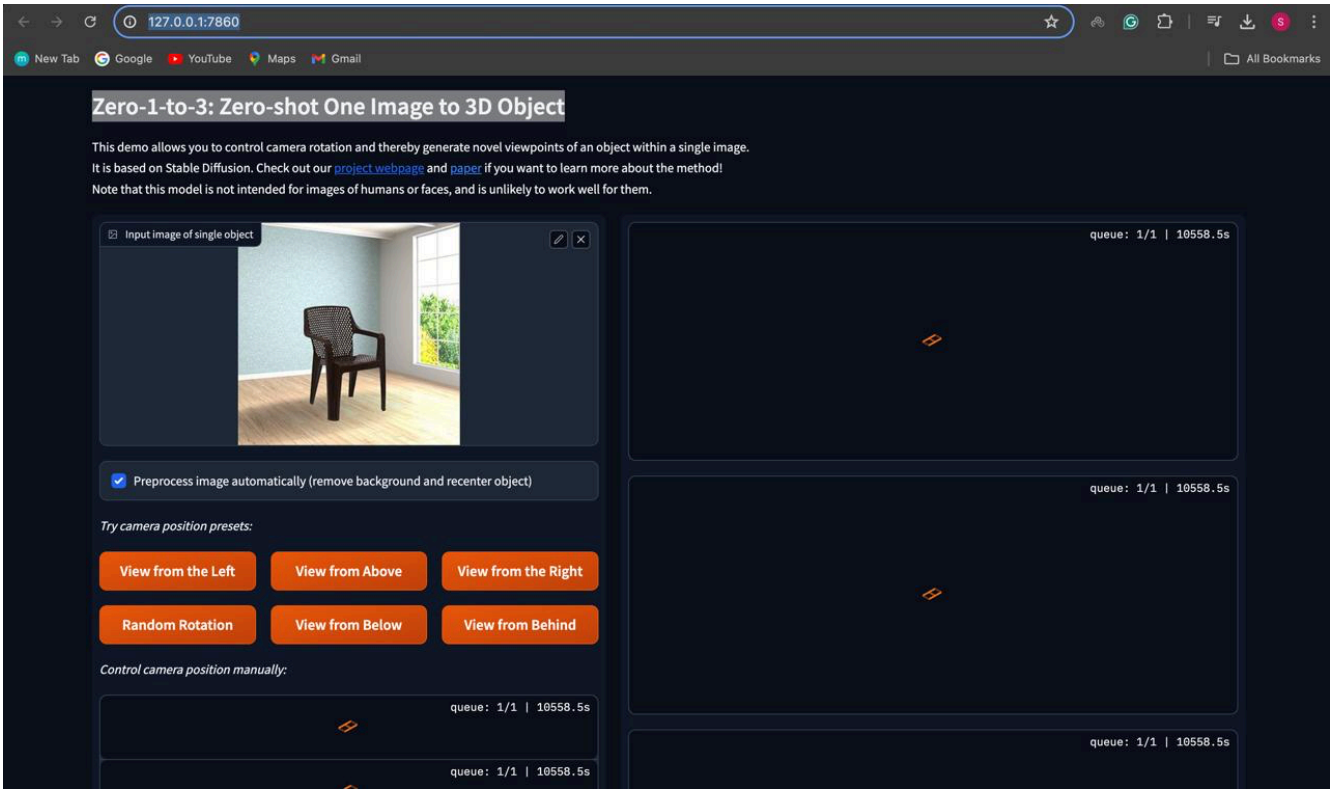
Planned Approach 1 :

After Segmentation of the object , using the model [Zero-1-to-3: Zero-shot One Image to 3D Object](#) and obtaining the required results.

Setback :

The model hosted online at https://huggingface.co/spaces/cvlab/zero123 does not have the option of uploading images, user can only choose from one of their images.

Upon running the gradio_app from the repo https://github.com/cvlab-columbia/zero123 locally the model takes forever to give results (Does not give any ,even after a very long time)



Also based on my understanding of the paper and interaction with the hosted app https://huggingface.co/spaces/cvlab/zero123

I realized that I needed the object on a white background for the model to work the best.

Therefore first an object removal pipeline was set and the object was separated out, to apply the rotations on and then they could be superimposed later.

Background without chair

Separated Chair

Finding a workaround

Since the zero1to3 needs very heavy computing to run , I was not able to run the model, if the proposed workflow is followed the process will work without any problem.

However to demonstrate results: I followed the proposed workflow to get different angles of the object using rotation of the image and then smartly (using bbox of where the object was present earlier the position of overlaying is decided)overlaying the objects on the background image.

One thing to note is that when an image is rotated or flipped the background is also flipped along with it and to avoid that I first separate the object from the background and then perform flips and rotations on it and later overlay the object in the background based on where it was present in the image earlier

Have a look at various intermediate steps during the process

Input Image

Background Image without Object

Chair Separated out

Chair flipped horizontally

Overlaying flipped chair on Background

Output 1

Output 2

Failed approach with inpainting using **stabilityai/stable-diffusion-2-inpainting**

Input Image **Output Image**

Output for text prompt : The same chair in the image but turned 180 degrees horizontally

Discussion on Background Inpainting

Some failed cases for good background inpainting:

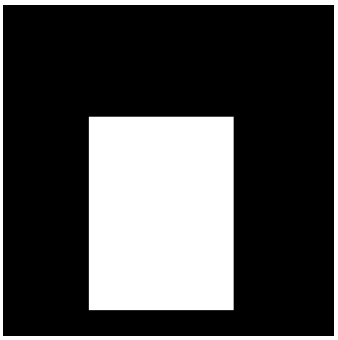
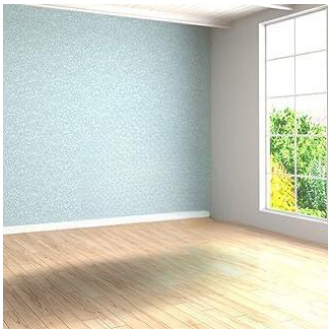
The inpainting of the background might have failed due to the object covering a significant area in the image and when the mask was given it was not able to correctly figure out background

Another observation :

You may have noticed that we give a rectangular mask instead of an actual segmentation mask for background inpainting the reason is when we give segmentation mask for the task we get outputs like this



Input **Mask** **Output**

The reason behind such results can be that when we segment we might miss out on some object pixels and the inpainting model since it depends on neighbouring pixels of mask we might end up getting bad results as not all neighbouring pixels are gone but in rectangular all the neighbouring pixels are gone