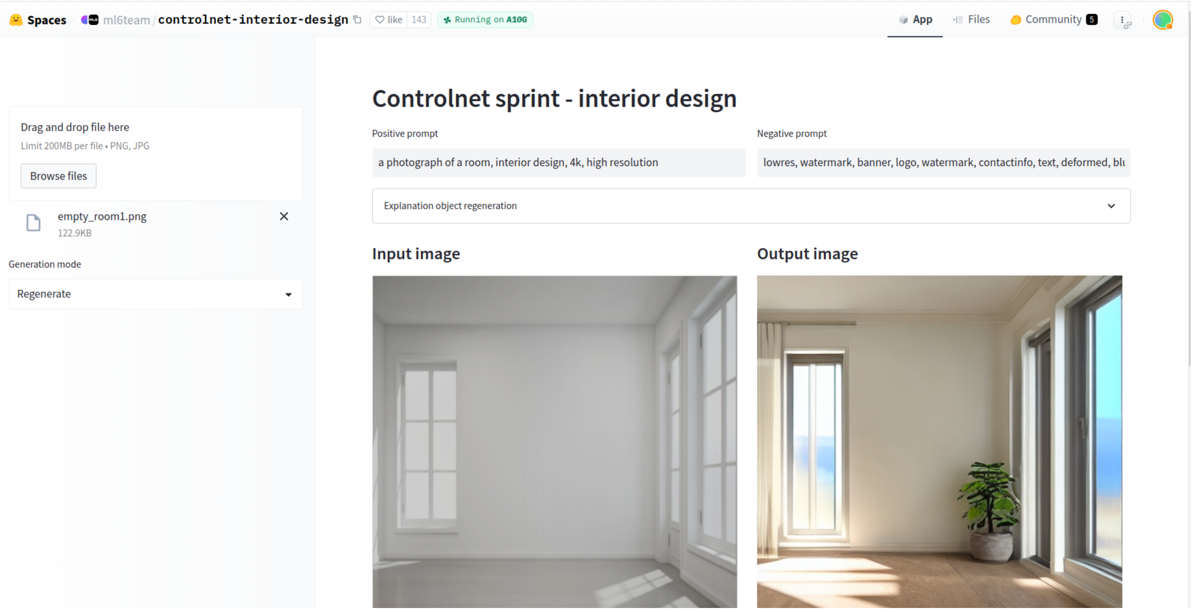
How HF space provided in challenge works

- HF space (for regenerate option) is like



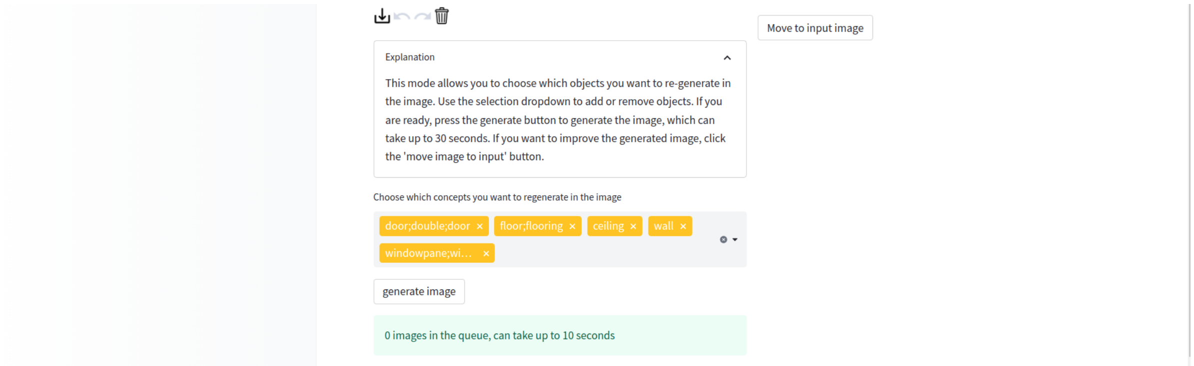
Controlnet sprint - interior design

empty_room1.png
122.9KB

Generation mode: Regenerate

Positive prompt: a photograph of a room, interior design, 4k, high resolution

Negative prompt: lowres, watermark, banner, logo, watermark, contactinfo, text, deformed, blı



Explanation

This mode allows you to choose which objects you want to re-generate in the image. Use the selection dropdown to add or remove objects. If you are ready, press the generate button to generate the image, which can take up to 30 seconds. If you want to improve the generated image, click the 'move image to input' button.

Choose which concepts you want to regenerate in the image: door;double;door, floor;flooring, ceiling, wall, windowpane;wi…

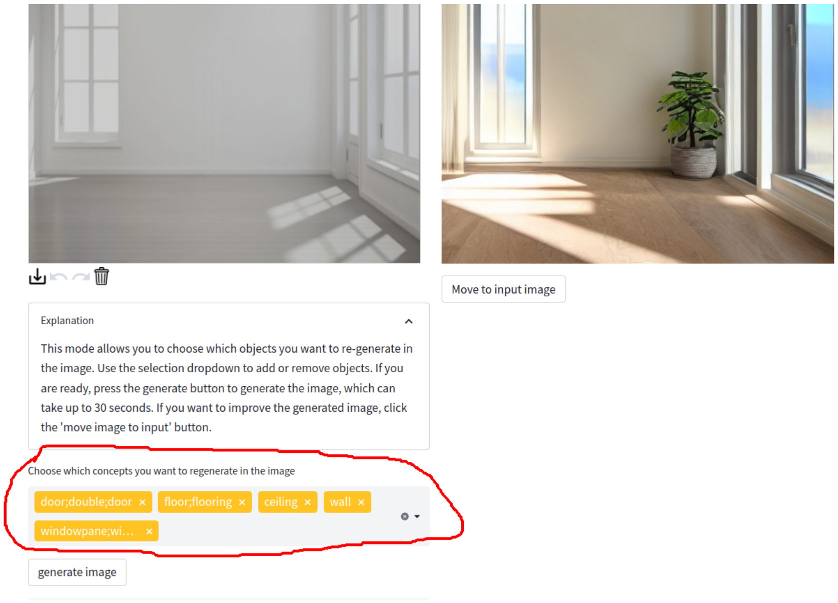
0 images in the queue, can take up to 10 seconds

- Segmentation from Original image is created

```
image_processor = AutoImageProcessor.from_pretrained("openmmlab/upernet-convnext-small")
image_segmentor = UperNetForSemanticSegmentation.from_
```

```
pretrained(
        "openmmlab/upernet-convnext-small")
```

- Mask is created (based on the areas we want to regenerate)



Explanation

This mode allows you to choose which objects you want to re-generate in the image. Use the selection dropdown to add or remove objects. If you are ready, press the generate button to generate the image, which can take up to 30 seconds. If you want to improve the generated image, click the 'move image to input' button.

Choose which concepts you want to regenerate in the image

door;double;door × floor;flooring × ceiling × wall × windowpane;wi… ×

```
mask = np.zeros_like(segmentation)
for color in chosen_colors:
    # if the color is in the segmentation, set mask to 1
    mask[np.where((segmentation == color).all(axis=2))] = 1
```

- Then controlnet is called

```
result_image = make_image_controlnet
(image=image,
 mask_image=mask,
 controlnet_conditioning_image=segmentation,
 positive_prompt=st.session_state['positive_prompt'],
 negative_prompt=st.session_state['negative_prompt'],
 seed=random.randint(0, 100000) # nosec
)
```

- Controlnet pipeline used is

  ```
  self.controlnet = ControlNetModel.from_pretrained
  ("BertChristiaens/controlnet-seg-room",
  torch_dtype=torch.float16)
  self.pipe =
  StableDiffusionControlNetInpaintImg2ImgPipeline.
  from_pretrained(
  "runwayml/stable-diffusion-inpainting",
      controlnet=self.controlnet,
      safety_checker=None,
      torch_dtype=torch.float16
  )
  ```

- Then, using this pipeline output image is generated

  ```
  generated_image = pipe(
          prompt=positive_prompt,
          negative_prompt=negative_prompt,
          num_inference_steps=50,
          strength=1.00,
          guidance_scale=7.0,
          generator=
  [torch.Generator(device="cuda").manual_seed(seed)],
          image=image,
          mask_image=mask_image,
          controlnet_conditioning_image=
  controlnet_conditioning_image,
      ).images[0]
  ```