Machine Learning Engineering Bootcamp
Capstone, Step 5: Data Wrangling
Student: Kenneth Fung
Date: 18 January 2025

1. INTRODUCTION:

In this step 5 of my Capstone project, I gathered a dataset of outdoor images to process using Photoshop into a dataset of images with embedded pictures and mirror reflections which are simulated portions of the original images. The objective is to create a dataset of images for self-supervised fine-tuning the Depth Anything v2 MDE model. The goal is to improve the pretrained Depth Anything model's performance in processing depth maps of images with flat pictures such as paintings, or mirror reflections.

2. Depth Anything v2 – Areas For Improvement:

The following are potential areas of improvement for the Depth Anything v2 MDE model:

A. Figure 1: This scene contains a painting of farm animals, as evidenced by the male and female figures behind the painting (hence it cannot be a window). The painting should have resulted in a constant depth assignment, and outlines of the animals have not have resulted.

B. Figure 2: There is a faint outline of the lady in the mirror. Since the mirror is flat, it should have resulted in a constant depth gradient instead of an outline of the lady. The image of the chess piece in the mirror fared better.

Figure 1. Depth Anything - Area for improvement: embedded pictures

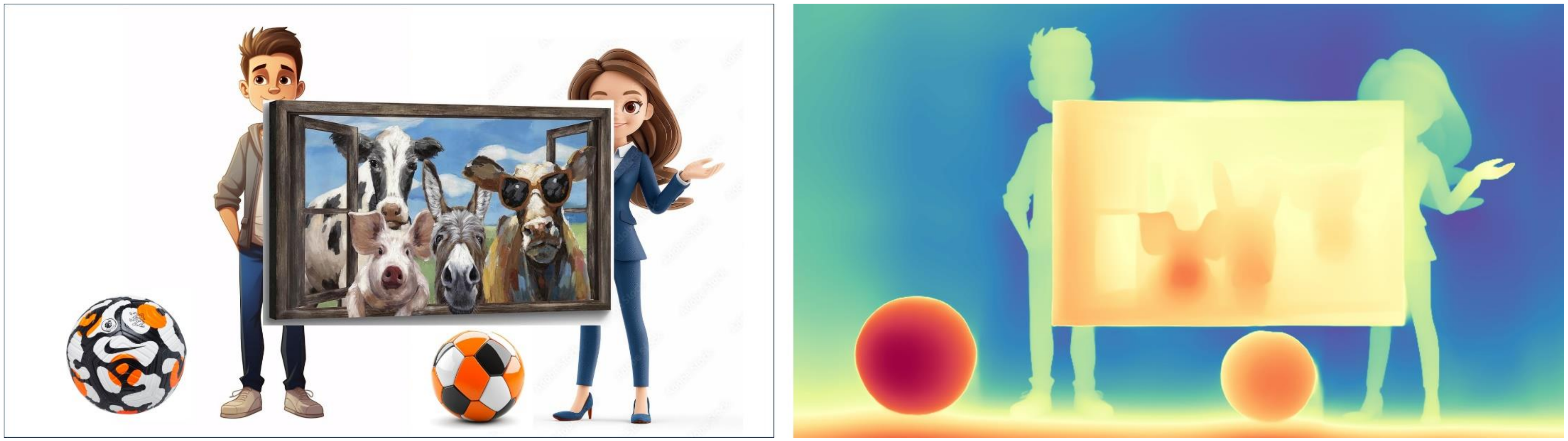
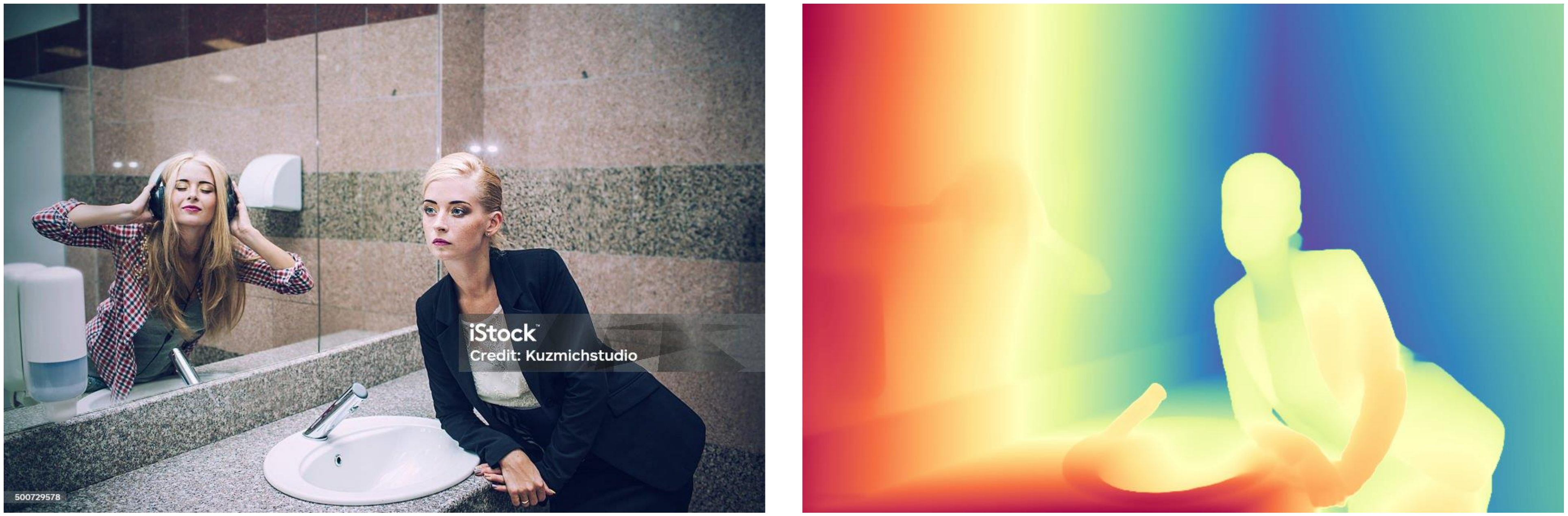


Figure 2. Depth Anything – Area for improvement: mirror reflections



3. ANALYSIS of Areas For Improvement

Pattern recognition for depth determination in monocular images are highly dependent on overlapping surfaces. For instance, if two non-overlapping objects with no overlapping background are perpendicularly placed onto parallel planes, and an image is captured where the camera angle is perpendicular to the planes, then it would be impossible to determine the relative depth of the objects. Similarly, if an object is "in front" of another in an image, a MDE model would determine that the object in front is closer to the camera, however would not know the relative depth between the object in front and the object behind. This is evident in figures 5 and 6. Figure 5 shows the output from Depth Anything on an image with a simulated painting, and figure 6 shows a Depth Anything output with a simulated mirror reflection. The two outputs show different depth maps for the objects in the background, although the depth map for the objects in the background relative to each other in each respective out is consistent. This may be a deficiency that is impossible to improve upon with an MDE model alone.

What should be correctable is the depth maps of the embedded pictures (paintings) and mirror reflections, as displayed in figures 5 and 6. In both figures, there is clearly a frame around the painting and the mirror reflection. These frames, along with the background, indicate the frames enclose flat images (not objects with relative depth) because they could not be windows. It is physically impossible, for instance, to have a window in front of the lady in figure 5, and show the lady's legs but block the rest of the lady. The frames would normally indicate a flat image within the frame to an observer, and most circumstances, an MDE model should have produced a depth map of only one tone or gradual tones with no distinguishable shapes for the flat image.

4. PLAN FORWARD

I downloaded a dataset of around 14,000 images from Mapillary[1]. This dataset did not require data wrangling to clean. I then used Photoshop to batch process these images to copy and paste a portion of the image onto the original image and then draw a frame around the pasted sub-image – brown for paintings and silver for mirror reflections. The process of self-supervised fine-tuning will be used to fine-tune the Depth Anything MDE model with the altered Mapillary dataset of images. Both methods of self-supervised fine-tuning and incremental training to improve the well-trained Depth Anything MDE model were considered, and self-supervised fine-tuning was chosen. The authors of Depth Anything v2 have provided a codebase[2] to fine-tune our Depth Anything V2 pre-trained encoder for metric depth estimation.

A. Gather a large (~ 14,000 images) dataset of images of consistent sizes and resolution, consisting of either indoor or outdoor scenes.

B. Use Photoshop to batch process the dataset to produce two subsets of images – one subset with embedded pictures such as paintings, and one subset with embedded mirror reflections of portions of the original images.

C. Use Depth Anything v2 to process the set of Mapillary images which was processed by Photoshop in step B. Use the arguments "--grayscale " and "—pred-only" to produce a predicted grayscale depth map

[1] https://www.mapillary.com/dataset/vistas
[2] https://github.com/DepthAnything/Depth-Anything-V2/tree/main/metric_depth

image only without the raw image. These gray scale images will be processed further in the next step to provide the ground truth depth maps to fine-tune Depth Anything.

D. Use Python and the OpenCV library to process the grayscale images from the previous step to identify the bounding rectangle in each image, then change all pixel values within the rectangle to the highest pixel value within the region bounded by the rectangle. The processed grayscale image would then be saved in a HDF5 format to be paired with the raw image as a paired input to the Depth Anything MDE model to train and fine-tune.

E. Follow this process[3][4]:

C1. Run Depth Anything on Your Dataset

- Use the pre-trained **Depth Anything** model to infer depth maps for all images in your dataset.
- Ensure your dataset is preprocessed correctly (e.g., resizing images to the required dimensions, normalizing values) to match the input requirements of the model.
- Save the generated depth maps as ground truth labels in a format compatible with the model's training pipeline.

C2. Evaluate the Generated Depth Maps

- Check the quality of the depth maps produced by **Depth Anything**.
    - Visualize a subset of the outputs.
    - Look for areas with significant noise or artifacts.
- If the quality is not consistent, consider:
    - Using additional post-processing techniques like edge-preserving filters or smoothing.
    - Combining depth maps with other sources (e.g., stereo depth, SfM outputs) for more reliable labels.

C3. Augment the Dataset

- Perform **data augmentation** to improve the robustness of the retraining process:
    - Flip, rotate, or scale the images and corresponding depth maps.
    - Add synthetic reflections, noise, or blur to mimic challenging conditions.
- Augmentation helps avoid overfitting to the biases of the pre-trained model.

[3] OpenAI. (2025). Explanation of self-supervised fine-tuning process. Retrieved from ChatGPT
[4] https://github.com/DepthAnything/Depth-Anything-V2/tree/main/metric_depth

---

C4. Fine-tune the Depth Anything Model

- Set up the training pipeline for **Depth Anything**:
    - Use the inferred depth maps as pseudo-labels for supervision.
    - Implement loss functions such as **L1 Loss**, **Huber Loss**, or **SSIM Loss** to compare predicted depth maps with pseudo-labels.
    - Use regularization techniques (e.g., smoothness losses) to enforce realistic depth continuity.
- Train with a lower learning rate initially to prevent catastrophic forgetting of the pre-trained knowledge.

C5. Validate on a Separate Dataset

- Split your dataset into training, validation, and test sets.
- Ensure that the model generalizes well to unseen images by monitoring validation metrics like **RMSE**, **Abs Rel Error**, and **Delta Accuracy**.

C6. Iterative Refinement

- Once the model is retrained, you can:
    - Use it to generate more accurate depth maps for your dataset.
    - Repeat the process iteratively for further refinement.
- Optionally, combine the predictions of the retrained model with the original pseudo-labels to improve robustness.

C7. Potential Issues and Considerations

- **Bias Propagation**: Since the pseudo-labels are generated by the same model, any inherent biases in the pre-trained model might persist.
    - Mitigation: Introduce additional high-quality ground truth depth maps (if available) to fine-tune alongside pseudo-labels.
- **Domain Shift**: If your dataset differs significantly from the pre-trained model's training data, results might be suboptimal.
    - Mitigation: Include domain-specific augmentations or a few real depth labels from your target domain.

Tools and Resources for Implementation

- **Frameworks**: PyTorch, TensorFlow, or any library compatible with Depth Anything.
- **Pre-trained Model Repository**: Ensure you have access to the original **Depth Anything** weights and inference scripts.

Visualization Libraries: Use tools like Matplotlib or OpenCV for depth map visualization and evaluation.

5. Using Photoshop to process Mapillary dataset:
I used Photoshop to batch process the images downloaded from Mapillary. The processing involved copying and pasting a portion of the image onto the original image and then drawing a frame around the pasted sub-image – brown for paintings and silver for mirror reflections. The photoshop scripts and sample processed images are available in my GitHub repository as indicated below. Figures 3-6 show samples of processed images, with figures 5-6 showing the outputs of the Depth Anything MDE model on two samples.

A. Mapillary dataset (mapillary-vistas-dataset_public_v1.2.zip):
https://www.mapillary.com/dataset/vistas

B. Photoshop script for adding embedded pictures (paintings) to original images:
GitHub: https://github.com/kentheman4AI/SB-Capstone-Project-Monocular-Depth-Estimation/blob/3edd44604f4ee6b7f730d72efdc2537226132bdd/Photoshop%20scripts/ResizeAndFrame.jsx

C. Photoshop script for adding mirror reflections to original images:
GitHub: https://github.com/kentheman4AI/SB-Capstone-Project-Monocular-Depth-Estimation/blob/3edd44604f4ee6b7f730d72efdc2537226132bdd/Photoshop%20scripts/MirrorImageProcessing.jsx

D. Sample Mapillary images processed with Photoshop:
GitHub: https://github.com/kentheman4AI/SB-Capstone-Project-Monocular-Depth-Estimation/tree/3edd44604f4ee6b7f730d72efdc2537226132bdd/data

Figure 3. Sample image of Mapillary image processed by Photoshop to add an embedded picture (painting)

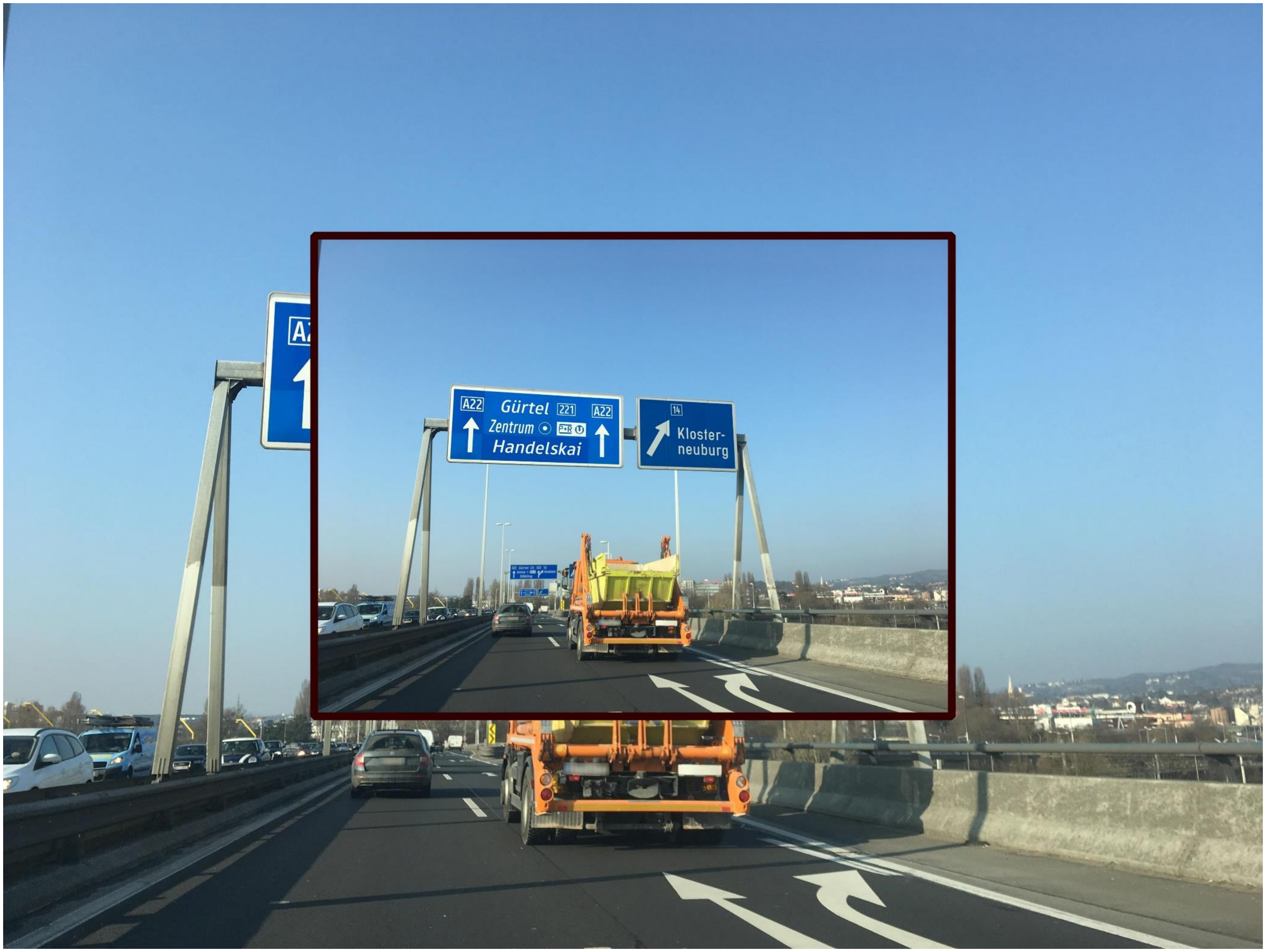


Figure 4. Sample image of Mapillary image processed by Photoshop to add a mirror reflection

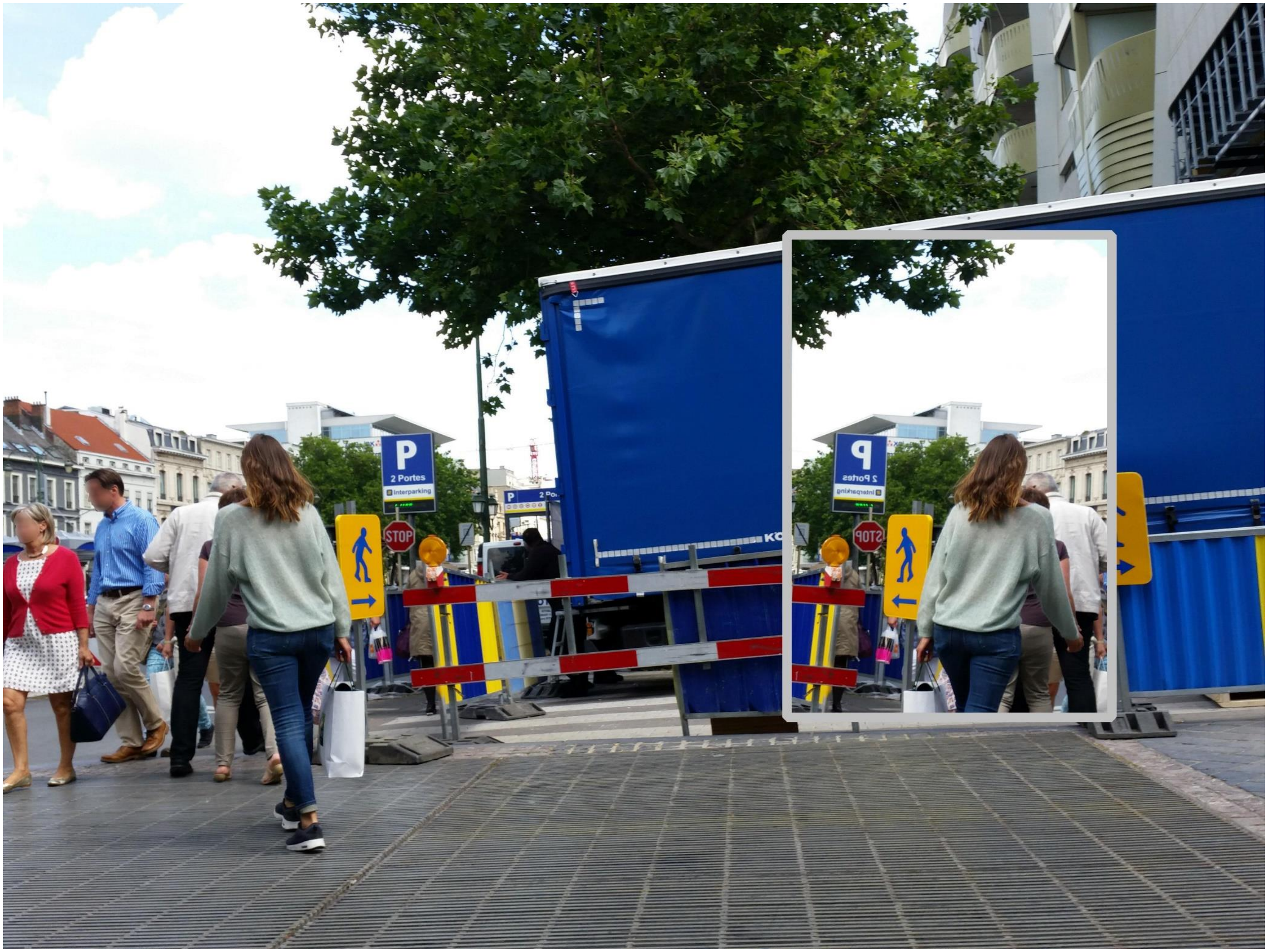


Figure 5. Depth Anything Outputs of Embedded Image

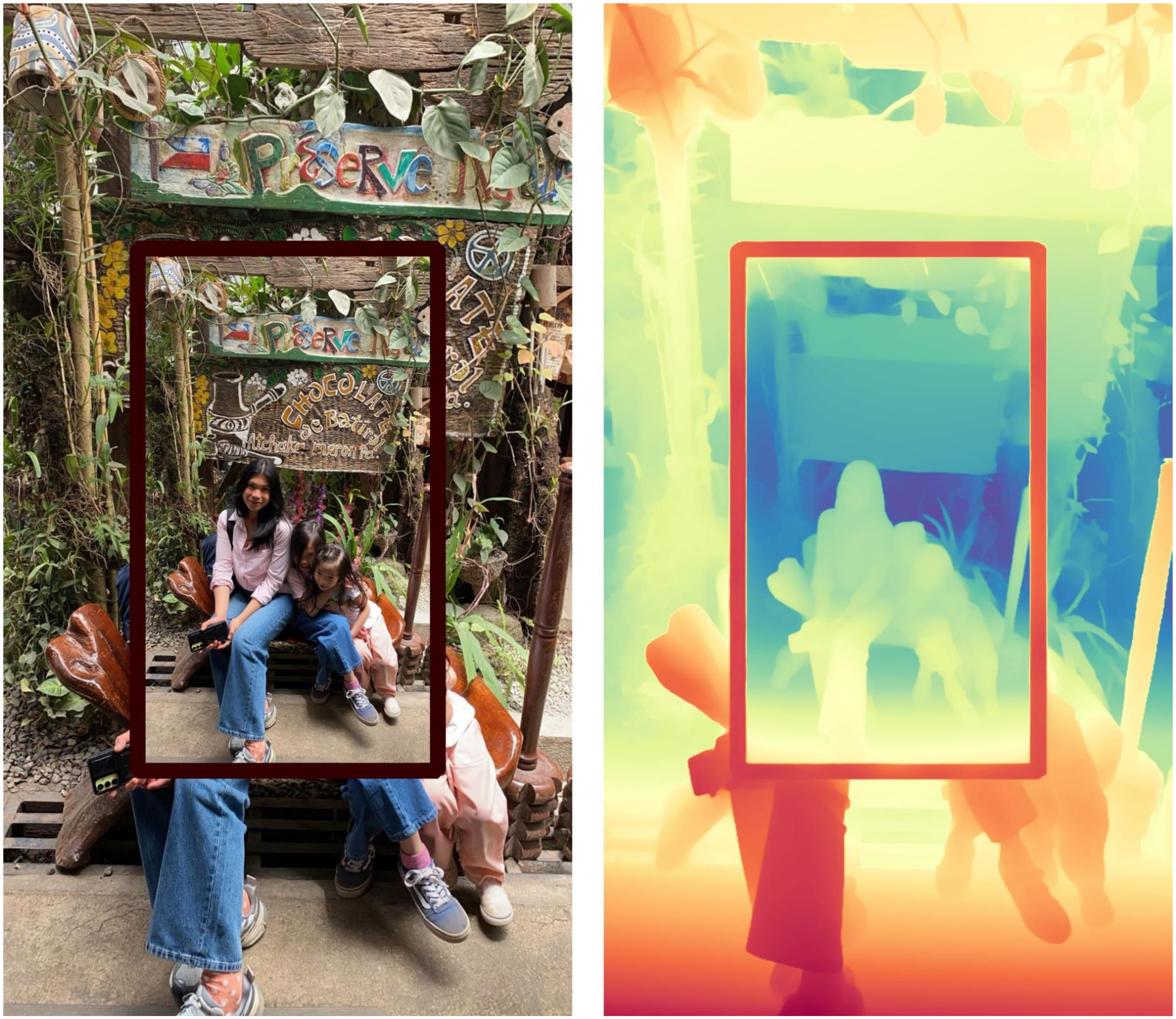


Figure 6. Depth Anything Outputs of Reflection