Object Property Detection Results Report

**Objective:**

Recognize an object and determine its physical properties, such as material, surface friction, dimensions, and approximate weight, to set initial parameters for a robotic gripper.

**Methodology:**

The project, *Object-Property-Detection*, involves:

1. **Image Capture and Segmentation:** Capturing images and segmenting objects within them using either batch\_yoloseg.py or rs\_yoloseg\_capture.py.
2. **Object Property Extraction Approaches:**
   * **Approach 1: Captioning + LLM**
     + Step 1: Generate a caption for the object using the Salesforce/BLIP image-captioning model (captioning.py).
     + Step 2: Extract object properties from the generated caption using Mistral 7B (ollama\_mistral.py).
   * **Approach 2: Direct Input to LLaVA**
     + Step 1: Pass the segmented object directly to LLaVA (ollama\_llava.py) for property extraction.

**GPU Memory Usage:**

* **Approach 1:**
  + *Captioning*: Salesforce/BLIP - 1.88 GB
  + *LLM (Mistral 7B)*: 4.1 GB
* **Approach 2:**
  + *LLaVA*: 4.1 GB

**Time Analysis (for a single image):**

**Image Given:** 

**Segmented Image (Yolo Seg) :**

****

* **Approach 1:**
  + Object Segmentation: < 0.1 sec
  + Image Captioning: 1.7540 sec
  + Mistral Query: 3.9017 sec
  + **Total Time:** ~5.76 sec
* **Approach 2:**
  + Object Segmentation: < 0.1 sec
  + LLaVA Query: 2.0388 sec
  + **Total Time:** ~2.14 sec

**Final Results:**

* **Approach 1:**
  + "Object name": Mouse
  + "Material": Fur, Plastic, Steel (for components)
  + "Coefficient of friction": Approximately 0.6 - 0.8 for fur on most surfaces, varies depending on surface and condition of the fur
  + "Dimensions (L x W x H in cm)": Approximately (10 x 5 x 3) cm
  + "Weight (in grams)": 25-100 grams
* **Approach 2:**
  + "Object name": Mouse
  + "Material": Plastic
  + "Coefficient of friction": 0.45
  + "Dimensions (L x W x H in cm)": 16 x 7.5 x 3.2
  + "Weight (in grams)": 98

**Prompt Used:**

The following prompt was used for both approaches: "Identify the object in the image and provide the following material properties. Respond only with the required values, no comments or notes:

* 'Object name': (one or two words)
* 'Material': (one or two words)
* 'Coefficient of friction': (numeric value)
* 'Dimensions (L x W x H in cm)': (3 numeric values separated by commas)
* 'Weight (in grams)': (numeric value)"

**Comparison of Approaches:**

| **Criteria** | **Approach 1: Captioning + LLM** | **Approach 2: Direct Input to LLaVA** |
| --- | --- | --- |
| **Workflow** | Image captioning, then property extraction via LLM | Directly queries LLaVA with the image for properties |
| **Time Efficiency** | ~5.76 seconds per image | ~2.14 seconds per image |
| **Detail in Output** | Provides richer details with more nuanced descriptions | More concise and specific responses |
| **GPU Memory Usage** | Total of ~5.98 GB (1.88 GB + 4.1 GB) | ~4.1 GB |
| **Complexity** | More complex, involves two separate models | Simpler, with a single model for extraction |

**Pros and Cons:**

* **Approach 1: Captioning + LLM**
  + **Pros:**
    - Richer output with multiple material descriptions.
    - Provides a range of values, which can be useful for uncertainty estimation.
  + **Cons:**
    - Higher processing time, leading to slower performance for real-time applications.
    - More memory consumption due to running two models.
    - Additional complexity in managing the interactions between captioning and property extraction models.
* **Approach 2: Direct Input to LLaVA**
  + **Pros:**
    - Faster processing time, making it suitable for real-time applications.
    - Uses less memory, which can be advantageous for deployments on systems with limited resources.
    - Simpler workflow, reducing the likelihood of errors during model interaction.
  + **Cons:**
    - Less detailed output, which may miss nuanced descriptions.
    - May provide overly simplified or approximate properties that lack precision in some cases.

**Summary:**

* **Efficiency:** Approach 2 (Direct input to LLaVA) is faster, taking around 2.14 seconds per image compared to 5.76 seconds for Approach 1.
* **Detail in Results:** Approach 1 provides more nuanced descriptions, including multiple materials and a range for friction. However, Approach 2 delivers more specific and concise properties.
* **Memory Usage:** Both approaches require significant GPU memory, but Approach 1 uses an additional 1.88 GB for captioning, potentially affecting scalability.
* **Use Case Recommendation:** Approach 1 is recommended for scenarios where richer descriptions are necessary, while Approach 2 is preferable for faster results in real-time applications.