**Báo cáo Counting & Finding**

**1. Giới thiệu**

Template matching là một phương pháp được sử dụng trong xử lý ảnh để tìm kiếm và so khớp một mẫu (template) trong một hình ảnh lớn hơn. Đây là một kỹ thuật hữu ích để xác định vị trí của các đối tượng có trong ảnh dựa trên mẫu cho trước. Trong bài toán tìm đồ vật, template matching thường kết hợp với thuật toán NMS (Non-Maximum Suppression) để loại bỏ các khung bao trùng lặp.

### **2. Template Matching với OpenCV**

OpenCV cung cấp hàm cv2.matchTemplate() để thực hiện template matching. Khi áp dụng, hàm này sẽ trả về một ma trận biểu thị mức độ tương đồng giữa template và các vùng khác nhau của ảnh nguồn. Dựa trên kết quả này, người dùng có thể xác định được vị trí mà template khớp nhất với ảnh.

Các bước thực hiện template matching:

* **Bước 1:** Chuẩn bị ảnh nguồn và ảnh template.
* **Bước 2:** Sử dụng cv2.matchTemplate() để tính toán ma trận tương đồng.
* **Bước 3:** Dùng hàm cv2.minMaxLoc() để tìm ra vị trí tương đồng tốt nhất giữa template và ảnh nguồn.

### **3. Thuật toán Non-Maximum Suppression (NMS)**

NMS là một kỹ thuật thường được sử dụng sau khi template matching nhằm loại bỏ các khung bao (bounding boxes) không cần thiết hoặc bị trùng lặp. Sau khi thực hiện template matching, ta có thể có nhiều vùng khớp, bao gồm các khung bao lớn, nhỏ và chồng lên nhau. NMS giúp giữ lại chỉ những khung bao có giá trị cao nhất và loại bỏ những khung trùng lặp không cần thiết.

Các bước thực hiện NMS:

* **Bước 1:** Sắp xếp các khung bao theo độ tương đồng (score) giảm dần.
* **Bước 2:** Loại bỏ các khung có độ chồng lặp (IoU - Intersection over Union) cao hơn ngưỡng xác định với khung có score cao nhất.
* **Bước 3:** Tiếp tục lặp lại cho đến khi không còn khung nào cần loại bỏ.

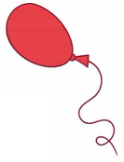
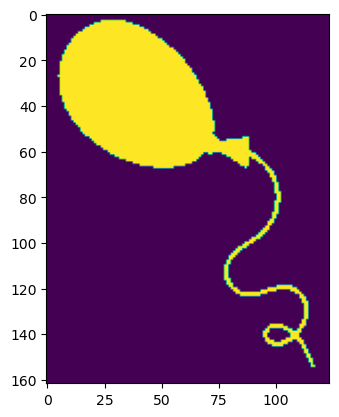
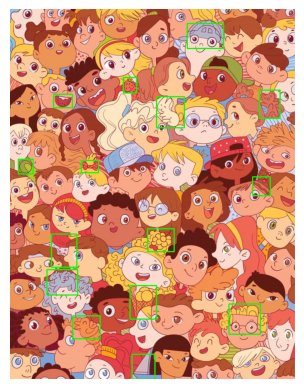
### **4. Ứng dụng trong bài toán tìm và đếm Objects**

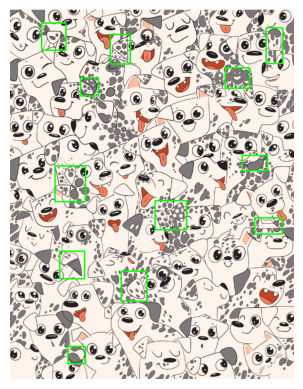
### **Task 1: Finding - Tìm kiếm đồ vật trong ảnh**

#### **Mô tả**

Trong task Finding, mục tiêu là tìm kiếm các đồ vật đã cho (template) từ danh sách và xác định vị trí của chúng trong ảnh gốc. Cách tiếp cận chính trong bài toán này là cắt các đối tượng từ danh sách (dùng làm template) và sau đó sử dụng thuật toán template matching để tìm vị trí tương ứng của chúng trong ảnh chính. Khi làm việc với nhiều tỷ lệ của template, ta sẽ chọn tỷ lệ có độ tương đồng cao nhất.

#### **Các bước chính**

* **Bước 1:** Đọc ảnh từ danh sách đồ vật.
* **Bước 2:** Tạo mask cho template (nếu cần), giúp tập trung vào vùng quan trọng trong ảnh, có tác dụng phân vùng giữa object và background (như hình dưới), để khi dùng matchTemplate tăng độ chính xác.
* 
* **Bước 3:** Tìm đối tượng trong ảnh gốc với các tỷ lệ khác nhau của template (scale) để xử lý sự khác biệt kích thước.
* **Bước 4:** Sử dụng hàm cv2.matchTemplate() để tính toán mức độ tương đồng (dùng phương pháp TM\_CCOEFF\_NORMED).
* **Bước 5:** Dùng cv2.minMaxLoc() để lấy vị trí có giá trị tương đồng cao nhất.
* **Bước 6:** Nếu giá trị tương đồng vượt qua ngưỡng đặt trước (threshold), đối tượng được coi là đã tìm thấy.
* **Bước 7:** Vẽ khung bao quanh vị trí tìm được trên ảnh gốc.



### **Task 2: Counting**

Trong task này, chúng ta sẽ đếm các đối tượng (object) thỏa mãn yêu cầu bằng cách nhận diện các đặc điểm riêng biệt của chúng. Đối với mỗi đối tượng, phương pháp nhận diện có thể khác nhau dựa trên đặc điểm hình học hoặc màu sắc của chúng.

#### **Subtask 1: Đếm mèo trong số cú mèo**

* **Phân biệt mèo và cú mèo**: Mèo và cú mèo có nhiều điểm tương đồng về hình dáng, tuy nhiên điểm khác biệt dễ nhận ra nhất là ở phần **mỏ/miệng**. Trong khi cú có mỏ nhọn, mèo có miệng phẳng hơn. Hình 1 là mèo còn hình 2 và 3 là cú. Ta có thể thấy phần mắt/tai có phàn tương đồng nhưng mỏ cú và miệng mèo có hình dáng khác nhau.

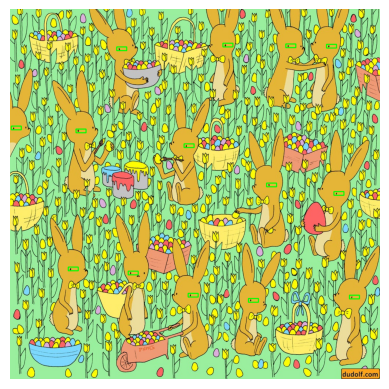


* **Phương pháp**: Ta có thể sử dụng template matching tập trung vào vùng miệng để đếm số lượng mèo. Bằng cách tạo template miệng mèo và cú, ta dùng phương pháp so khớp mẫu để xác định vị trí của mèo trong ảnh.



#### **Subtask 2: Đếm thỏ 1**

* **(Đếm dựa trên số mắt)**: Một đặc điểm dễ nhận diện của thỏ là mắt đen, nhỏ. Trong trường hợp này, ta có thể tạo một template cho mắt thỏ và đếm số lượng mắt trong ảnh. Có 1 lưu ý là có những con thỏ có chỉ có 1 mắt (góc nhìn ngang). Để giải quyết vấn đề này ta có thể sử dụng một đặc điểm là các con thỏ đều chỉ có một mắt cách xa phần đầu. Sử dụng template này  ta có thể đếm được số lượng thỏ.

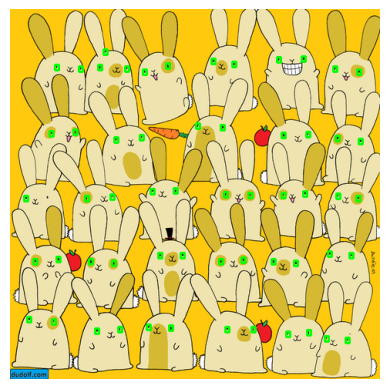


**Subtask 3: Đếm thỏ 2**

* **Hai hướng tiếp cận**:
  1. **Dựa trên số miệng thỏ**: Miệng của thỏ có hình dáng đặc biệt và dễ nhận ra, có thể sử dụng template cho miệng và đếm số lượng miệng. Tuy nhiên ta có thể nhận thấy một số con thỏ có miệng khác nhau nên hướng tiếp cận này không khả quan.



* 1. **Dựa trên số mắt**: Sử dụng phương pháp tương tự như ở **Thỏ 1**, ta đếm số lượng mắt thỏ để xác định số lượng thỏ. Tuy một số chỗ vẫn match sai (có thể do threshhold/ảnh template chưa chuẩn) nhưng kết quả vẫn tốt hơn match bằng miệng.

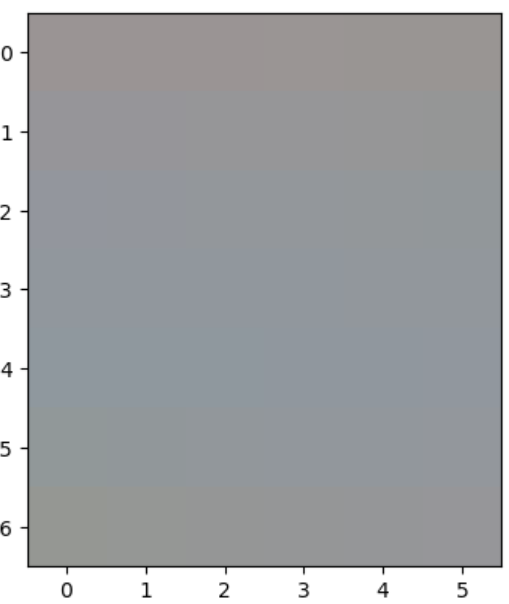


#### **Subtask 3: Đếm số đôi ủng**

* **Nhận diện hình dạng ủng**: Các đôi ủng thường có hình dạng giống nhau, điều này giúp cho việc nhận diện chúng dễ dàng hơn. Có thể tạo template cho đôi ủng và sử dụng template matching để đếm số đôi ủng trong ảnh. Ta có thể thấy có một số đôi ủng chưa được match (do bị khuất một số đồ vật), điều này cho thấy ảnh template của chúng ta càn được crop sát viền hoặc nên crop một phần thay vì toàn bộ đôi ủng.

#### **Subtask 5: Đếm số con chuột**

* **Nhận diện chuột qua màu lông xám**: Chuột là đối tượng khó nhận diện nhất vì chúng có hình dạng khác nhau. Tuy nhiên, các con chuột đều có một phần lông xám đặc biệt, với mức độ xám khác biệt so với các đối tượng khác trong ảnh. Bằng cách xác định các vùng xám đặc trưng này, ta có thể đếm số con chuột.



* **Phương pháp**: Phương pháp sử dụng cửa sổ trượt (sliding window) để tính toán sự khác biệt màu sắc giữa một mẫu (template) và các vùng tương ứng trong ảnh. Tại mỗi vị trí cửa sổ, ta tính khoảng cách chuẩn (norm) giữa vùng ảnh và template để đo sự khác biệt. Kết quả được chuẩn hóa vào khoảng [0, 1], sau đó áp dụng ngưỡng (threshold) để tìm các vị trí có khả năng chứa đối tượng. Cuối cùng, các khung bao được nhóm lại (group rectangles) để loại bỏ sự chồng lấp và nhận diện các đối tượng chính xác hơn.

