Universidad Nacional de Rosario

Trabajo práctico Nº 03

Procesamiento de Imágenes

Tecnicatura Universitaria en Inteligencia Artificial

Periodo: Diciembre 2025

Integrantes:

- de Brito, Nicolás - Legajo D-4416/4
- Giacone, Agustín - Legajo G-5917/1
- Taborda, Matías - Legajo T-3158/5

Índice

Introducción

Este informe detalla la resolución de los problemas propuestos, aplicando diversas técnicas estudiadas en la asignatura. Para el ejercicio, se explican detalladamente los pasos seguidos, algunos obstáculos encontrados y las soluciones implementadas.

Se incluyen recursos visuales, como imágenes, para facilitar la comprensión de los resultados y, en ciertos casos, ejemplos prácticos del funcionamiento de los programas desarrollados. El trabajo se basa en un repositorio de GitHub que contiene un archivo README con las instrucciones de ejecución, dos archivos Python con las soluciones y el material didáctico proporcionado por la cátedra.

Problema - Cinco dados

Contexto del problema

A partir de un conjunto de 4 videos correspondientes a tiradas de 5 dados, el trabajo consiste en el desarrollo de un algoritmo capaz de detectar automáticamente el momento en el que los dados quedan estáticos para poder marcarlos con su bounding box correspondiente, asignarles un nombre, mostrar el valor del mismo y generar un video de salida. El objetivo de este trabajo es desarrollar un algoritmo que, a partir de cuatro videos de tiradas de cinco dados, sea capaz de identificar automáticamente el instante en que los dados se detienen. Una vez detectado el reposo, el algoritmo deberá realizar las siguientes tareas:

- Marcar cada dado con su *bounding box* correspondiente.
- Asignar un nombre a cada dado.
- Mostrar el valor obtenido por cada dado.
- Generar un video de salida con estas anotaciones.

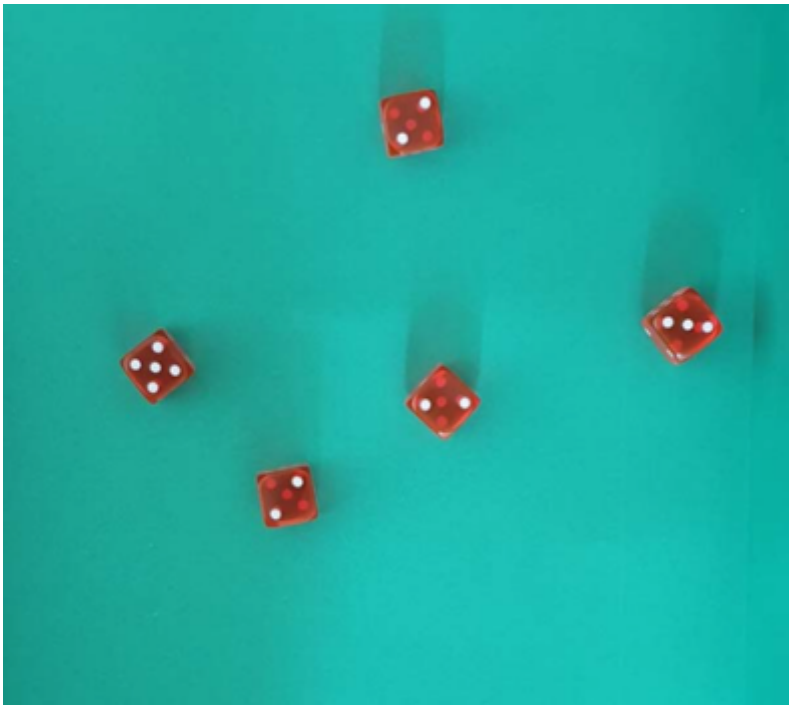

Figura 1: Crop de un frame con los dados estáticos.

Resolución del problema

Identificación de elementos - Mascara

El primer paso consistió en generar una máscara para identificar los objetos presentes en la imagen. Para ello, se definió la función `generar_mascara()`, la cual toma un frame como entrada y retorna una máscara binaria: fondo negro y objetos en blanco.

La estrategia principal se basó en el color de los dados. Los videos analizados contienen tiradas de dados de color rojo intenso sobre un fondo verde, lo que proporciona un fuerte contraste.

Aprovechando esta diferencia cromática, se convirtió el frame al formato HSV para manipular los canales H (Tono) y S (Saturación).
Sabiendo que el color rojo se sitúa en los extremos del canal H (valores muy bajos o muy altos), se aplicó un filtro a los píxeles con valores inferiores a 10 o superiores a 170.
Adicionalmente, dado que los dados eran de un rojo intenso, se decidió filtrar el canal S, seleccionando valores por encima de 100 para asegurar la saturación.
Finalmente, ambas máscaras obtenidas se combinaron y se aplicó una operación morfológica de clausura para refinar la forma de los objetos detectados.

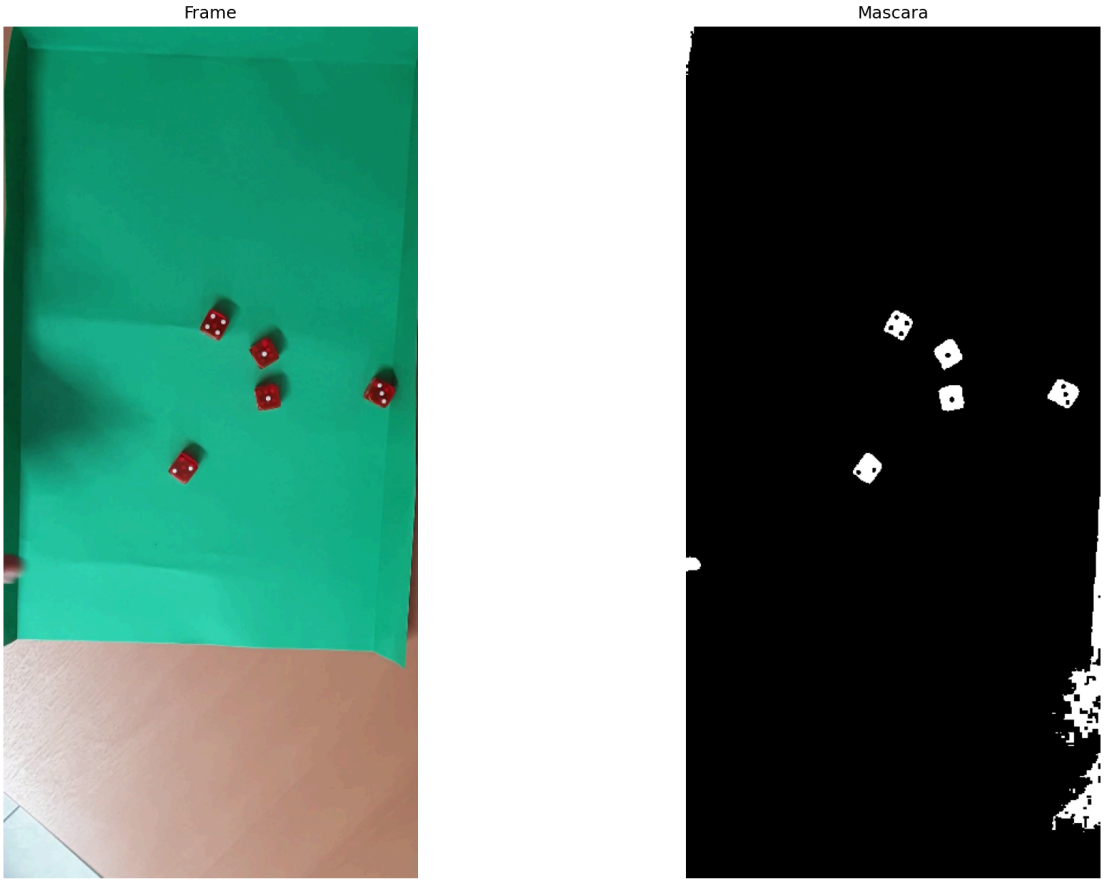

Figura 2: Comparación de un frame con su máscara.

Identificación de los dados

A partir de esta máscara, se procedió a la detección de componentes. Por cada elemento identificado, se invocaba la función `es_dado()`, la cual retorna:

- **-1:** Si el elemento detectado no es considerado un dado.
- **0:** Si el elemento es considerado un dado, pero su valor no pudo ser detectado.
- **>=1 y <=5:** Si el elemento es un dado y representa el valor detectado.

El propósito central de esta función es doble: primero, identificar cuáles elementos son dados y, segundo, determinar el valor de aquellos que lo son.

Para la identificación, se utilizó la función `findContours`, aprovechando la estructura jerárquica que ofrece `RETR_TREE`. La premisa es que un dado presenta un contorno externo (su borde) y un número de contornos internos (hijos del contorno externo) que representan su valor.

Tras la aplicación de la función, se seleccionaron únicamente los contornos que no tenían padres. A estos contornos se les calculó el área con `contourArea`, el perímetro con `arcLenght` y, finalmente, la circularidad.

Considerando que un cuadrado ideal tiene una circularidad aproximada de 0.785, se establecieron dos filtros:

1. **Circularidad:** Se filtraron elementos con una circularidad superior a 0.5 (para incorporar un margen debido a la irregularidad de los contornos reales).
2. **Área:** Se filtraron aquellos elementos cuyo ratio de área se encontraba en un rango de 0.0001 a 0.0005 con respecto al área total del frame.

Si el contorno detectado cumplía con ser un contorno padre y clasificaba como dado según los filtros anteriores, se procedía a contar la cantidad de contornos hijos asociados a dicho contorno principal.

Figura 3: Comparación de un elemento reconocido como dado con la detección de sus contornos.

La imagen de ejemplo ilustra un elemento que superó exitosamente todos los filtros. El contorno principal (padre) se distingue en color rojo, mientras que los contornos anidados (hijos) se presentan en verde.

A medida que se detectaban los dados íbamos guardando la info del bounding box del elemento junto al valor del dado.

Detección de dados estáticos

Para determinar el momento en que los dados permanecían estáticos, se implementó una comparación frame a frame de su posición. Inicialmente, se verificaba que se hubieran detectado cinco dados, la misma cantidad tanto en el cuadro actual como en el anterior.

Una vez confirmada la cantidad, se invocaba la función `calcular_distancias()`, cuyo objetivo es medir la distancia entre los dados de ambos cuadros. Esta función, como primer paso, ordena los dados de cada frame, primero por la coordenada *y* y luego por la coordenada *x* (de arriba hacia abajo y de izquierda a derecha). Posteriormente, calcula el centroide de cada dado y mide la distancia entre el centroide del dado *i* del cuadro anterior y el centroide del dado *i* del cuadro actual.

El concepto principal se centra en mantener una pequeña variación en la distancia entre frames consecutivos. Experimentalmente, se estableció que si la suma de estas distancias era menor o igual a 5, se incrementaría en uno un contador de frames consecutivos. Si este contador alcanzaba un mínimo de 7, se consideraba que los dados estaban estáticos.

A partir de ese punto, se procedió a dibujar el *bounding box* de cada dado en el *frame* respectivo, incluyendo un nombre que lo identificara y su valor correspondiente.

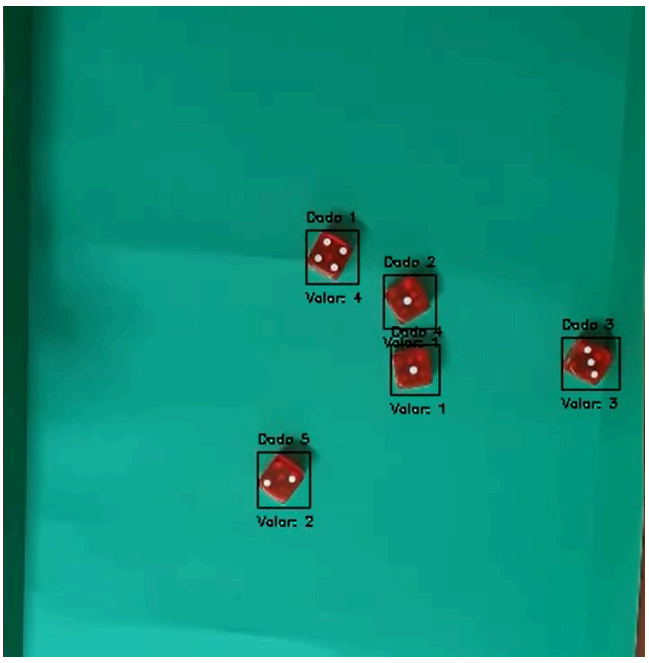

Figura 4: Frame de dados identificados con su boundingbox, nombre y valor.

Finalmente se genera un archivo de video como salida del proceso, homónimo al archivo de entrada, alojado en la ubicación *videos/output*.

Además, se muestra por consola un mensaje similar al siguiente:

```
Iniciando procesamiento tirada_1.mp4...
----------------------------------------
Los dados detectados son los siguientes:
----------------------------------------
Dado 1. Valor: 4
Dado 2. Valor: 1
Dado 3. Valor: 3
Dado 4. Valor: 1
Dado 5. Valor: 2
----------------------------------------
Fin procesamiento tirada_1.mp4
****************************************
```

Figura 5: Información mostrada por consola ante el procesamiento de una imagen.

Conclusión

En este trabajo se desarrolló un algoritmo capaz de analizar videos de tiradas de dados y detectar automáticamente el momento en el que estos dejan de moverse. A partir de ese instante, el sistema identifica cada dado, lo marca en la imagen, le asigna un nombre y muestra el valor obtenido, generando finalmente un video de salida con toda esta información visible.

El enfoque elegido aprovechó una característica clave del problema: el fuerte contraste de color entre los dados y el fondo. Esto permitió simplificar la detección de los objetos y reducir la complejidad general del proceso. A partir de ahí, se trabajó sobre la forma y la estructura típica de un dado para diferenciarlo de otros elementos y reconocer su valor de manera automática.

La detección del momento en que los dados quedan estáticos resultó un punto central del trabajo. Comparando la posición de los dados entre frames consecutivos y permitiendo pequeñas variaciones, se logró una solución práctica y bastante robusta para determinar cuándo los dados habían dejado de trasladarse sobre la mesa.

Finalmente, el sistema desarrollado cumple con los objetivos planteados y demuestra cómo, combinando técnicas básicas de procesamiento de imágenes con criterios simples pero bien definidos, es posible resolver un problema concreto de forma efectiva.