Universidad de Buenos Aires
Facultad de Ciencias Exactas y Naturales
Departamento de Computación

Seguimiento de Objetos en Secuencias de Imágenes RGB-D

Tesis presentada para optar al título de
Licenciado en Ciencias de la Computación

Mariano Bianchi

Director: Francisco Roberto Gómez Fernández

Buenos Aires, 2014

SEGUIMIENTO DE OBJETOS EN SECUENCIAS DE IMÁGENES RGB-D

Acá iría el abstract en español (aprox. 200 palabras).

Palabras claves: español, abstract, acá (no menos de 5).

OBJECT TRACKING USING RGB-D IMAGE SEQUENCES

Escribir acá el abstract IN ENGLISH ;) (aprox. 200 palabras).

Keywords: Escribir, ENGLISH, acá (no menos de 5).

AGRADECIMIENTOS

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Fusce sapien ipsum, aliquet eget convallis at, adipiscing non odio. Donec porttitor tincidunt cursus. In tellus dui, varius sed scelerisque faucibus, sagittis non magna. Vestibulum ante ipsum primis in faucibus orci luctus et ultrices posuere cubilia Curae; Mauris et luctus justo. Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos himenaeos. Mauris sit amet purus massa, sed sodales justo. Mauris id mi sed orci porttitor dictum. Donec vitae mi non leo consectetur tempus vel et sapien. Curabitur enim quam, sollicitudin id iaculis id, congue euismod diam. Sed in eros nec urna lacinia porttitor ut vitae nulla. Ut mattis, erat et laoreet feugiat, lacus urna hendrerit nisi, at tincidunt dui justo at felis. Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos himenaeos. Ut iaculis euismod magna et consequat. Mauris eu augue in ipsum elementum dictum. Sed accumsan, velit vel vehicula dignissim, nibh tellus consequat metus, vel fringilla neque dolor in dolor. Aliquam ac justo ut lectus iaculis pharetra vitae sed turpis. Aliquam pulvinar lorem vel ipsum auctor et hendrerit nisl molestie. Donec id felis nec ante placerat vehicula. Sed lacus risus, aliquet vel facilisis eu, placerat vitae augue.

Índice general

1. INTRODUCCIÓN

En la actualidad, las posibles aplicaciones de métodos de seguimiento o tracking son muchas y van desde el uso en la industria hasta juegos de consola. Un ejemplo de ello es la fabricación de barcos y autos mediante el uso de robots. Estas tareas se caracterizan por la necesidad de posicionar de manera precisa una herramienta sobre una pieza de trabajo. A través del uso de métodos de tracking se puede conocer la ubicación y pose de la pieza que se desea utilizar con respecto a la ubicación de la cámara y de esta forma saber cómo posicionar la herramienta necesaria para trabajar sobre la pieza en cuestión.

Otra área en donde se utiliza tracking de objetos intensamente es para la generación de estadísticas deportivas, por ejemplo, durante un partido de fútbol, tanto de jugadores como de un equipo. Las posibles aplicaciones en este contexto son mucho más amplias, como por ejemplo análisis de tácticas, verificación de las decisiones del árbitro, resúmenes automáticos de un partido, etc.

Actualmente existen sensores de profundidad que en conjunto con una cámara RGB pueden ser utilizados para detectar y seguir a una o más personas en tiempo real. De esta manera, mediante un sistema que procese las imágenes RGB-D de estos sensores, las personas puedan utilizar su cuerpo y sus movimientos para interactuar naturalmente con un dispositivo.

La utilización de sensores RGB-D se ha popularizado en los últimos años, cobrando un gran interés científico el estudio de aplicaciones y métodos capaces de procesar y entender la información que los mismos proveen.

La información de profundidad obtenida por un sensor RGB-D es un dato fundamental que nos posibilita encontrar la distancia de un objeto con respecto al sensor pudiendo recuperar su información tridimensional (3D) junto a su textura RGB en tiempo real (30 cuadros por segundo). El video RGB-D

que se obtiene provee una gran ayuda al mejoramiento y desarrollo de nuevas técnicas de procesamiento de imágenes y video ya conocidas. En particular, en esta tesis nos enfocamos en el seguimiento de objetos en secuencias de imágenes RGB-D.

mbianchi: Esto va en la introduccion? (ex "trabajo relacionado")

En el artículo [PLW11] se implementan las tres etapas de un sistema de seguimiento nombradas anteriormente. Cada una de estas etapas es abordada de distintas maneras según la literatura actual. La etapa de entrenamiento consiste en obtener una representación tridimensional del objeto al cuál se pretende seguir. En el artículo [DC99] se utiliza un entrenamiento off-line que consiste en obtener un modelo CAD (computer-aided design) del objeto que se desea seguir. Luego, en el artículo [PLW11] se presenta una etapa de entrenamiento novedosa que se realiza de manera on-line, en donde utiliza un marcador conocido para definir las coordenadas de los objetos y calibrar la cámara.

La etapa de detección tiene como objetivo obtener la ubicación del objeto a seguir en un frame dado. En el artículo [PLW11] utilizan el método propuesto en [HLI+10] para detección de objetos en imágenes 2D y lo extienden para estimar la pose 3D. Otros métodos conocidos en la literatura son los propuestos en [Bru09, KRTA13].

La etapa de seguimiento 3D cuadro a cuadro es la más importante y de la que depende el éxito o fracaso de todo el sistema de seguimiento. En el artículo [PLW11] utilizan el algoritmo "Iterative Closest Point" (ICP) propuesto en [Zha94, BM92], refinando el resultado con datos de bordes tomados durante la fase de entrenamiento. El método utilizado por [DC99] se basa en la detección de bordes para realizar el seguimiento frame a frame.

El objetivo principal de esta tesis es la implementación, estudio y evaluación de un sistema de seguimiento de objetos en secuencias de imágenes RGB-D de objetos tridimensionales con forma conocida previamente que se pueda aplicar a datos/escenas obtenidas a través de sensores de profundidad de bajo costo (Kinect, XTion, etc.).

mbianchi: Falta escribir: aportes e importancia, breve resultados y conclusiones y organizacion de la tesis por capitulo

2. SISTEMA DE SEGUIMIENTO

Un sistema de seguimiento se puede dividir en tres etapas bien definidas:

1. Entrenamiento
2. Detección
3. Seguimiento cuadro a cuadro

La etapa de entrenamiento consiste en obtener una representación del objeto al cuál se pretende seguir. Para llevarla a cabo se puede utilizar un patrón (template) ya conocido o aprenderlo de imágenes capturadas del mismo objeto. Luego se utiliza en la detección para ubicar la representación del objeto dentro de una imagen cualquiera. Una vez conocido el template no se requiere de una nueva ejecución del entrenamiento.

La segunda etapa, la de detección, radica en encontrar dentro de un frame del video al objeto en cuestión utilizando el método de detección deseado, valiéndose de la información obtenida en la etapa de entrenamiento. Esta etapa se ejecuta, con el propósito de encontrar en la imagen el objeto a seguir, al comienzo del sistema de seguimiento y cuando el seguimiento cuadro a cuadro falla. Dado que la etapa de detección suele ser la más costosa en términos de desempeño computacional es deseable que se ejecute la menor cantidad de veces posible. Esta etapa es de suma importancia ya que de ella va a depender la eficacia final de todo el sistema.

Finalmente, la tercera etapa consiste en seguir cuadro a cuadro el objeto detectado en la etapa anterior. Es decir, teniendo la ubicación del objeto en un cuadro de video se desea identificar la posición del mismo objeto en el siguiente frame. Esta etapa es la más importante ya que es la que se ejecuta en cada frame del video. La eficiencia del método de seguimiento es lo que determinará que todo el sistema de seguimiento se consiga realizar eficientemente. Si la técnica de seguimiento tiene una efectividad baja, es

decir, no logra identificar la nueva posición del objeto en el siguiente cuadro, se debe volver a la etapa de detección cuyo desempeño computacional es mayor.

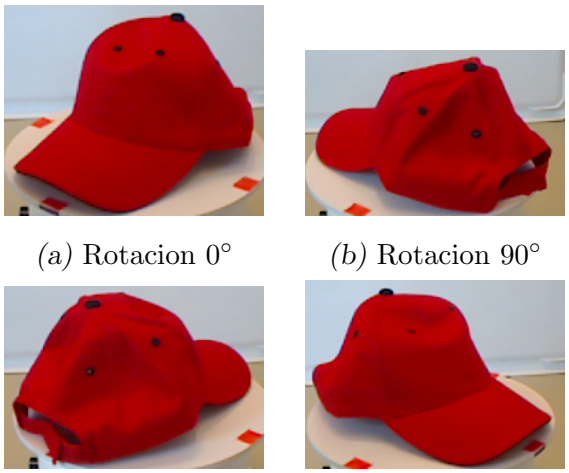
Tomando como base estas etapas, proponemos distintos métodos para cada una de ellas tanto para RGB como para D y RGB-D. La primera etapa del sistema puede ser prescindible si contamos con el modelo RGB-D del objeto a seguir y una cámara calibrada. Este es el caso de estudio de esta tesis, ya que, con el propósito de poder evaluar cuantitativamente el seguimiento de objetos en secuencias de imágenes RGB-D, utilizamos la base de datos descripta en la sección 3.

2.1. Método propuesto RGB

En esta sección explicaremos como se implementó el sistema de seguimiento para RGB basándonos en las etapas explicadas previamente.

2.1.1. Entrenamiento

La etapa de entrenamiento en este método consta de tomar de la base de datos cuatro templates del objeto que se desea seguir con sus respectivas máscaras. Cada uno de los templates corresponde a una pose distinta. Primero se elige de manera arbitraria una de las varias escenas por objeto que tiene la base de datos. Una vez elegida la escena se toman cuatro templates de manera que se cubran las distintas caras del objeto. Si se toma la primer imagen de la escena como la imagen con rotación 0 el resto de las imagenes se eligieron de manera tal que cada una difiera de la anterior en 90 grados. En la figura 2.1 se pueden ver los distintos templates tomados de esta manera para un objeto de la base de datos. De esta manera con cuatro templates es suficiente para cubrir todas las caras del objeto. Una vez almacenados estos templates y sus respectivas máscaras se pasa a la etapa de detección.



(a) Rotacion 0° *(b)* Rotacion 90°

(c) Rotacion 180° *(d)* Rotacion 270°

Fig. 2.1: Templates de una gorra tomados de la base de datos

2.1.2. Detección

Dada la información provista por la base de datos, aprovechada durante el entrenamiento, resulta natural utilizar como método de detección el algoritmo de *Template matching*.

mbianchi: ¿Hay que explicarlo?

La implementación utilizada es la que se encuentra presente en la librería *OpenCV* para C++. Pero aplicar este método de manera directa no es posible. El problema principal es que la escena de donde se obtuvieron los templates y máscaras del objeto a buscar no es la misma escena que la que se utiliza para verificar el comportamiento del sistema. Esto significa que las poses del objeto tomadas en la etapa del entrenamiento pueden ser completamente distintas a las poses del objeto en la escena en donde se aplica el sistema de seguimiento. Además, la distancia de la cámara al objeto en la escena de búsqueda varía constantemente y difiere de la distancia entre ambos en la escena de donde se capturó el modelo del objeto. Por este motivo también va diferir el tamaño del objeto en cada escena.

Teniendo en cuenta estos problemas decidimos realizar varias pasadas del algoritmo de template matching. En primer medida se iba modificando

el template y máscara utilizado como parámetro del mismo, aprovechando los templates obtenidos en la etapa anterior. Con esto se atacaba de mejor manera

2.1.3. Seguimiento

2.2. Método propuesto en D

2.2.1. Alignment prerejective

2.2.2. Iterative Closest Point (ICP)

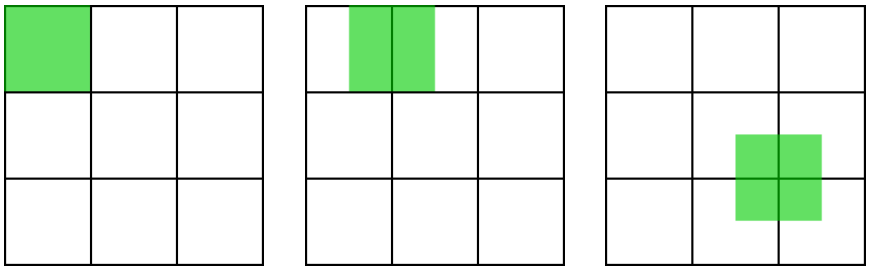
Para esta primera etapa existen varias posibilidades distintas que van desde utilizar una única nube de puntos hasta generar un modelo completo del objeto 3D alineando todas las nubes de puntos disponibles en la base.

mbianchi: La tarea de generar un modelo completo excede el tema de estudio de esta tesis. Además el modelo resultante se utilizara únicamente para la etapa de detección por lo que se optó por el método más simple que es tomar una nube de puntos cualquiera del objeto a seguir como modelo 3D.

pachi: Quedó afuera porque nos enfocamos en tracking

Por estos motivos se decidió elegir una nube de puntos fija para cada objeto, en particular, la que posee rotación 0. Otra posibilidad es elegir de todas las nubes de puntos de cada objeto presentes en la base alguna cualquiera al azar. Esto resulta más realista pero sería un problema al momento de analizar los resultados ya que se agregaría una variante azarosa y se deberían correr muchas veces la misma prueba para lograr un análisis más acertado.

Para la segunda etapa, la de detección, se utilizó el método descripto en la sección 2.2.1 refinando el resultado con ICP. La elección del mismo se realizó luego de correr varias pruebas que corroboraran la factibilidad del mismo. En estas pruebas también se observó que si la región donde se buscaba el objeto era lo suficientemente pequeña la búsqueda era más robusta.



(a) Primer frame de búsqueda

(b) Segundo frame de búsqueda

(c) 19° frame de búsqueda

Fig. 2.2: Se busca en cada cuadrante de la grilla y en los recuadros del mismo tamaño que cubren los bordes de la grilla principal

pachi: Mejor más chica?

Teniendo en cuenta esto se pensó en una variante para la detección que utilice el método elegido. Esta tiene como primer paso obtener el alto y el ancho del modelo del objeto a seguir y escalarlos según un valor, llamado **DETECTION_FRAME_SIZE**. Considerando estos valores dividimos la escena en cuadrantes de ese tamaño y corrimos el método de detección en cada cuadrante. Con el fin de detectar el objeto cuando el mismo se extiende sobre dos o más regiones, la búsqueda se hizo utilizando un marco que recorre todos los cuadrantes y sus uniones, como puede observarse en el gráfico 2.2. Notar que la división por cuadrantes solo se realizó en los ejes “x” e “y” y no en el eje de profundidad ya que las pruebas preeliminares dieron buenos resultados de esta manera y hacer eso implicaba agregarle complejidad algoritmica al método. La detección se corre en cada uno de estos cuadrantes y pueden suceder varias cosas:

- No se encontró el objeto en ningún cuadrante: en este caso el algoritmo indica que el objeto no se encuentra en el frame
- Se encontró el objeto en un cuadrante
- Se encontró el objeto en varios cuadrantes: el algoritmo devuelve la mejor posición encontrada según un puntaje de buena alineación devuelto

por el algoritmo *alignment prerejective*.

Si la detección es positiva, se refina la alineación corriendo ICP entre el modelo del objeto transformado por el método "alignment prerejective" y el cuadrante de la escena donde fue encontrado el mismo. Con el objetivo de comenzar el seguimiento en las mejores condiciones posibles, se intentan tomar los puntos del objeto buscado pertenecientes a la escena. Esto se realiza porque se asume que el objeto se va modificando cuadro a cuadro, ya sea por movimientos de la cámara o del objeto. Una de las formas para obtener los puntos del modelo del objeto en la escena es utilizando un *k-dtree*[1].

pachi: Explicar un poco que es un kdtree

Se arma un *k-dtree*con los puntos provenientes del modelo alineado y se filtran uno a uno los puntos de la escena que se encuentren cerca de al menos un punto del modelo en un cierto radio de distancia. Este valor de radio es uno de los parámetros explorados durante las pruebas, llamado **LEAF_SIZE**. Los puntos que surjan de esta búsqueda son los considerados encontrados en la escena. Para que el algoritmo de búsqueda considere exitosa la detección, la cantidad de puntos filtrados de la escena debe ser mayor o igual al 50 % de los puntos del modelo original. Si todas estas etapas son superadas con éxito, se considera que el objeto fue encontrado y se pasa a la siguiente etapa de seguimiento. Si cualquiera de estos pasos fallara, se comienza nuevamente con la etapa de detección en el siguiente frame.

La tercera y última etapa BLABLABLA

[1] AGREGAR REFERENCIA A KDTREE

2.2.3. Entrenamiento

2.2.4. Detección

2.2.5. Seguimiento

2.3. Método propuesto en RGB-D

2.3.1. Entrenamiento

2.3.2. Detección

2.3.3. Seguimiento

3. BASE DE DATOS RGB-D

Durante el desarrollo de este trabajo se utilizaron secuencias con información de ground truth de imágenes RGB-D anotadas para aplicar los métodos estudiados y tener una referencia para hacer comparaciones y sacar conclusiones sobre su eficacia. Las escenas fueron tomados del trabajo [LBRF11] en donde se creó una base de objetos y escenas. Esta base cuenta con varias escenas. Cada una de ellas consta de varios frames RGB con su respectiva información de profundidad. Además, la base provee información frame a frame de qué objetos aparecen y cuál es su ubicación en el plano RGB.

Figura con ejemplos de las escenas, por ejemplo: 2 filas con varias columnas cada una en donde haya frames RGB arriba y sus equivalentes en depth abajo

Por otra parte, la base provee imágenes RGB-D de los objetos presentes en las escenas antes mencionadas con el objetivo de obtener su representación 3D.

pachi: Comentar como fueron adquiridas la resolución, longitud y ¿?

Cada una de estas imágenes es acompañada además por una máscara que segmenta al objeto buscado y la información de profundidad (nube de puntos) del objeto segmentado. Para tomar estas imágenes los objetos fueron posados en una base circular giratoria y manteniendo la cámara en una posición fija se tomaron muestras con cierta regularidad cubriendo toda la circunferencia de cada objeto. Esto se hizo además desde distintas alturas permitiendo apreciar la profundidad del objeto y así obtener una mejor descripción del mismo.

Los objetos elegidos para esta base se organizaron de una manera jerárquica tomada de las relaciones hiperónimo/hipónimo de WordNet. Cada objeto

pertenece a una clase de objetos y hay varias instancias por cada clase. Por ejemplo, en la categoría "taza" existen varias instancias diferentes, que se corresponden simplemente a distintas tazas ya sea por forma o por color.
mbianchi: No hice pruebas usando varias instancias de un mismo objeto...

Existen distintas escenas que contienen a los objetos mencionados y en cada escena se combinan distintas clases de objetos y distintas instancias de la misma clase. De esta manera la base otorga la posibilidad de verificar algoritmos capaces de identificar instancias de objetos particulares o familias de objetos según la clasificación antes mencionada.

3.1. Parte de la propuesta que quedó afuera

La detección se realizó utilizando [BKK+13] y corrigiendo con ICP. Cosas a escribir:

- qué sucedió al tratar de detectar en toda la escena
- cómo se hizo para dividir la escena en partes y detectar en cada una

La utilización del algoritmo ICP [Zha94, BM92] para realizar el seguimiento resulta natural e intuitiva. Por ello, es que en esta tesis se estudiará el algoritmo ICP y sus variantes [EBW04, SHT09], con el fin de evaluar cómo sus parámetros afectan cuantitativamente al sistema de seguimiento y la performance computacional del mismo. Asimismo, se evaluará la adaptabilidad del filtro de Kalman [WB95] para seguimiento de objetos 3D en imágenes RGB-D con posibilidad de desempeño en tiempo real. El filtro de Kalman es un filtro muy popular y estudiado extensivamente en la literatura [JU97, WVDM00] debido a su gran desempeño para realizar seguimiento en imágenes 2D. Por lo tanto, su aplicación en seguimiento de objetos 3D resulta de especial interés.

3.2. Elección de parámetros

Método	coffee_mug_5			cap_4			bowl_3		
	% Over.	STD	% foll.	% Over.	STD	% foll.	% Over.	STD	% foll.
Bhatta. ch. verde	38.52	32.50	85.53	54.37	18.52	87.80	64.95	43.44	40.00
Correl. ch. verde	21.68	31.54	90.79	49.88	11.07	97.56	46.84	44.30	60.91
Bhatta. x canal	33.76	20.11	94.74	49.95	14.01	95.12	10.96	22.78	97.27
RGB y HSV	39.32	25.19	92.11	56.69	21.39	82.93	73.32	40.70	30.91

Tab. 3.1: Corregir valores!

4. RESULTADOS

4.1. Experimentación

Con el objetivo de elegir los métodos definitivos que usamos en cada etapa de cada sistema de seguimiento se realizaron variadas pruebas. En esta sección explicaremos cuales fueron estas pruebas, cómo se eligieron los valores para los distintos parámetros de cada método y mostraremos los resultados de los métodos elegidos, tanto para RGB como para D. También analizaremos el funcionamiento del sistema de seguimiento RGB-D.

Para lograr una buena comparación entre métodos de tracking, tanto para RGB como para D, se utilizó como método de detección el método ideal. El mismo consta simplemente de tomar los datos provistos por el ground truth para los frames en donde se requería correr una detección. En el caso de RGB los datos son tomados directamente desde el ground truth. Como la base de datos solo provee la ubicación del objeto en RGB en el caso de la detección en D estos datos se tomaban como punto de partida y luego se los refinaba tomando los datos del modelo 3D del objeto a buscar para correr AP e ICP y finalmente filtrar los puntos de la escena del objeto usando kdtree.

4.1.1. Elección del método de seguimiento RGB

Durante el proceso de selección de métodos se corrieron pruebas preeliminares para elegir aquellos que mejor se adaptaran a las escenas y objetos elegidos para este fin. Una vez hecho un primer filtro, se exploraron algunos parámetros de cada método para obtener los mejores valores de cada uno y luego hacer una selección final entre aquellos que mejor se desempeñaran. En esta sección explicaremos cuales fueron los métodos probados y cómo llegamos a elegir el definitivo, previamente explicado en la sección 2.1.

En un comienzo se tuvieron en cuenta dos métodos distintos para el segui-

miento RGB. El primero utilizaba la técnica de SURF para obtener keypoints y features y luego se utilizaba un comparador para hacer matching frame a frame. El segundo método consistía en extraer el histograma de la imagen RGB y utilizar un comparador de histogramas para hacer el seguimiento. Estos métodos se probaron con una escena tomada por una cámara web siguiendo distintos objetos. La misma no estaba anotada y sólo se utilizó para realizar pruebas preeliminares.

Luego de correr varias pruebas manuales se concluyó que el método que mejor se desempeñaba era el basado en histogramas por lo que se pasó a explorar distintas soluciones basadas en este método utilizando ahora las escenas tomadas de la base de datos. Este método tenía distintas variables a explorar:

- Esquema de colores: RGB o HSV
- Canales a utilizar de cada esquema
- Cantidad de bines por canal
- Método de comparación de histogramas: Bhattachayyra, Chi-Squared, Correlation, Intersection
- Umbral para la comparación frame a frame
- Umbral para la comparación modelo-frame

Para decidir qué esquema de colores, qué canales y cuántos bines por cada canal se iban a utilizar se eligieron dos objetos de la base con sus respectivas escenas y se tomaron muestras de cada uno junto con muestras de distintas texturas de la escena o del mismo objeto pero en donde se lo veía de manera parcial. Una vez tomadas las muestras se eligió una como template/modelo del objeto y se realizó una comparación entre el modelo y las distintas muestras tomadas de la escena y se ordenaron de mejor a peor matching. Esto se hizo fijando un método de comparación de histogramas. Un ejemplo de esto se puede ver en la figura 4.1. Esto se hizo con el objetivo

Fig. 4.1: Ordenando muestras según comparación por chi-squared, tomando los canales RGB, 16 bines para el canal rojo y 4 para el verde y el azul. El modelo utilizado es un template de la taza sacado de la escena.

de saber qué combinación de esquema, canales y bines por canal clasificaba mejor a las muestras.

De este análisis surgió que las mejores combinaciones de esquema, canales y bines para estos ejemplos fueron utilizar el canal verde en el caso de RGB con unos 60 bines, los 3 canales RGB con 8 bines por canal

mbianchi: Probé con 60 bines por canal y mejoró el de la taza considerablemente. Hacer bien las pruebas con 60 bines por canal.

y los canales SV del esquema de colores HSV, con 8 y 16 bines respectivamente. Una vez reducido el espectro de posibilidades comenzamos a realizar pruebas con el fin de definir para cada uno de estos esquemas qué método de comparación de histogramas convenía utilizar y con qué valor de umbral se obtenian mejores resultados. Los mejores resultados que obtuvimos se pueden observar en las tablas 4.1, 4.2 y 4.3. Para cuantificar la efectividad y robustes del tracking de cada algoritmo elegido decidimos evaluar las siguientes variables:

Objeto	% promedio de overlap	overlap STD	% veces seguido	Falsos Positivos	Falsos Negativos
Taza	29.69	35.24	86.84	0	0
Gorra	55.23	18.08	87.80	0	0
Bowl	66.60	42.51	39.09	0	0

Tab. 4.1: Mejores resultados usando comparación de histogramas por Bhattachayyra para el canal verde con 60 bines

- Taza de falsos positivos: cantidad de veces que el algoritmo reporta haber encontrado el objeto cuando en realidad no está en la imagen
- Taza de falsos negativos: cantidad de veces que el algoritmo no encuentra el objeto cuando en realidad el mismo está en la imagen
- Promedio de área solapada: promedio de solapamiento de área[1] entre el objeto reportado por el algoritmo y el ground truth durante toda la escena
- Desviación estándar del área solapada
- Porcentaje de veces seguido: de todas las veces que el objeto aparece en la escena obtenemos el porcentaje de veces que el algoritmo de seguimiento fue exitoso

4.1.2. Evaluación del tracking RGB

A continuación se muestran los resultados del algoritmo de seguimiento elegido para las imágenes RGB con los valores de los parámetros ya fijados. Para todas las pruebas se eligieron tres objetos distintos que aparecen en dos escenas, todos sacados de la base de datos indicada en la sección 3.

[1] Ver http://pascallin.ecs.soton.ac.uk/challenges/VOC/voc2011/workshop/voc_seg.pdf, página 7, Evaluation Metric

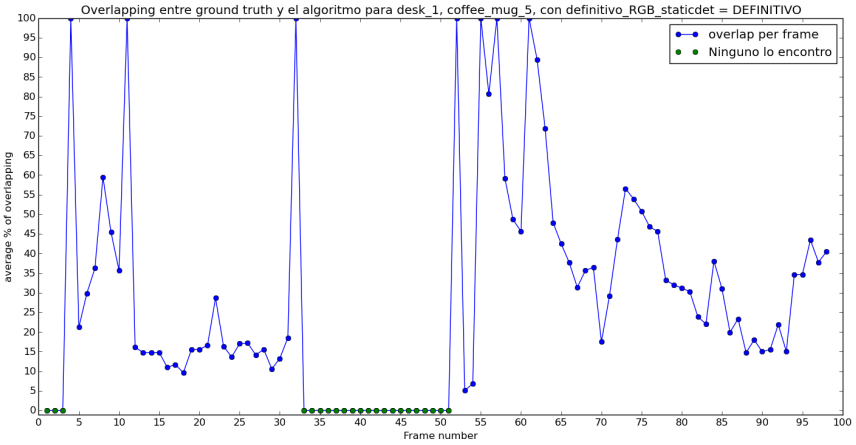
Objeto	% promedio de overlap	overlap STD	% veces seguido	Falsos Positivos	Falsos Negativos
Taza	24.69	33.94	88.16	0	0
Gorra	50.44	10.83	97.56	0	0
Bowl	50.68	42.37	60.00	10	0

Tab. 4.2: Mejores resultados usando comparación de histogramas por Correlation para el canal verde con 60 bines

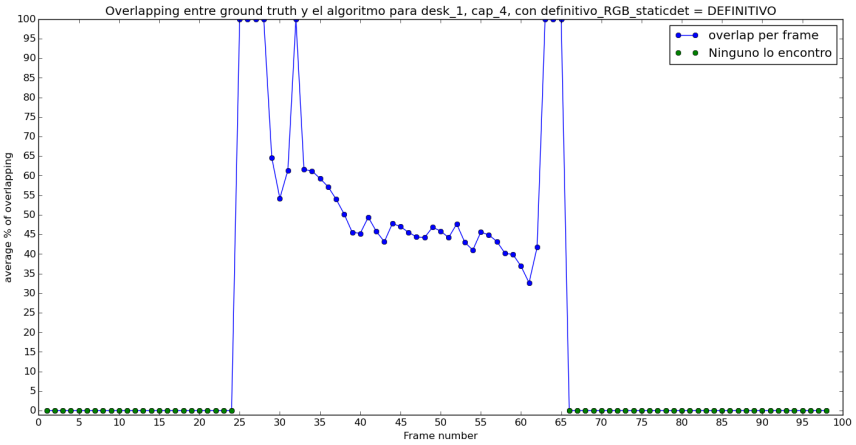
Objeto	% promedio de overlap	overlap STD	% veces seguido	Falsos Positivos	Falsos Negativos
Taza	36.75	26.20	90.79	0	0
Gorra	57.93	21.77	80.49	0	0
Bowl	74.13	40.24	30.91	0	0

Tab. 4.3: Mejores resultados combinando comparación de histogramas por Bhattachayyra para RGB y para SV (del esquema HSV)

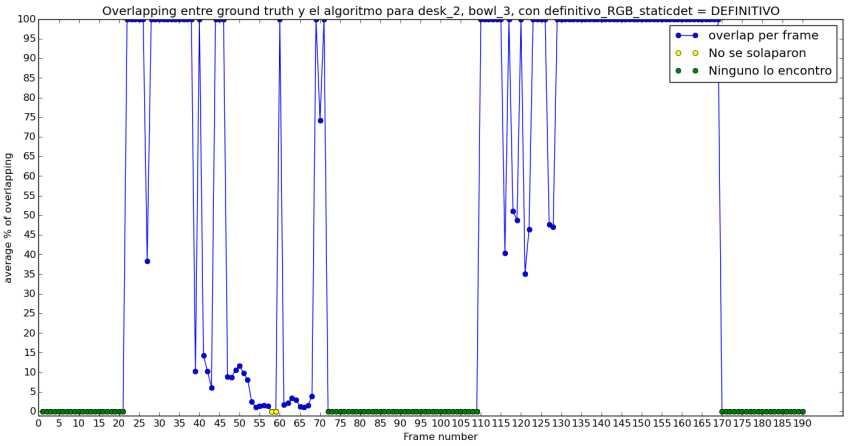
Como podemos ver en la tabla 4.4 el tracking se comporta de manera muy diversa dependiendo del objeto que se esté analizando. En los casos de la taza y la gorra el algoritmo es exitoso en la mayoría de los casos, 90 % y 80 % respectivamente. De todas maneras si se observa el promedio de solapamiento vemos que se comporta mucho mejor en el caso de la gorra. Creemos que esto se debe a la marcada diferencia de color entre la gorra y el fondo de la imagen. Esto no sucede en el caso de la taza en el que repetidas veces el color de fondo varía entre colores y tonos similares a los de esta lo que provoca que la comparación de histogramas no sea robusta. En el caso del bowl el promedio de solapamiento es alto pero se debe a que el porcentaje de veces que se siguió al objeto es bajo. Como el algoritmo de detección es el ideal, cuantas más veces se usa la detección mejor es el porcentaje de solapamiento. Este análisis está hecho en una escena distinta a la escena de la taza y la gorra. Notamos que en esta escena los cambios en la luminosidad y la coloración son muy notorios lo que afecta negativamente al algoritmo.



(a) Seguimiento frame a frame para la taza



(b) Seguimiento frame a frame para la gorra



(c) Seguimiento frame a frame para el bowl

Fig. 4.2:

mbianchi: Descripcion

Objeto	% promedio de overlap	overlap STD	% veces seguido	Falsos Positivos	Falsos Negativos
Taza	36.75	26.20	90.79	0	0
Gorra	57.93	21.77	80.49	0	0
Bowl	74.13	40.24	30.91	0	0

Tab. 4.4: Resultados del tracking RGB utilizando como detección los valores sacados de la base de datos

En la figura 4.2 se intenta visualizar mejor el comportamiento del algoritmo para cada objeto en las distintas escenas. En cada gráfico se muestra para cada escena y por cada frame el porcentaje de solapamiento entre el área del objeto reportada por el algoritmo y la indicada por el ground truth. Los puntos que están en 0 de color verde indican que el objeto no fue encontrado y que eso coincide con el ground truth, como es el caso de los gráficos 4.2a y 4.2b. En el gráfico 4.2c se ven dos puntos en 0 de color amarillo. Esto indica que el algoritmo reporta haber seguido al objeto pero que el área no se solapa con el área del ground truth. Para los tres gráficos, todas los frames cuyo área es igual a 100 % se corresponde con las veces que el algoritmo de detección fue corrido, es decir, cuando falló el seguimiento.

Se puede ver en el gráfico 4.2a que el algoritmo de seguimiento reporta un área que se solapa entre un 15 % y un 50 % en la mayoría de los casos. En el gráfico 4.2b la mayoría oscila entre un 35 % y un 50 % y en el gráfico 4.2c se observa que la mayoría se encuentra entre un 1 % y un 10 %. Una hipótesis es que el algoritmo no funciona correctamente cuando los objetos tienen poca textura y esto empeora si existen objetos cercanos cuya textura sea similar a la del objeto que se está buscando. Este sería el caso de la taza y del bowl. Ambos son objetos con poca textura de color blanco que pueden camuflarse con otros objetos de la escena.

mbianchi: No estaría bueno indicar el porcentaje de solapamiento promedio del algoritmo de seguimiento sin tener en cuenta las detecciones????

4.1.3. Método de seguimiento en D

El método de seguimiento elegido para profundidad es ICP, explicado en la sección 2.2.2. En esta sección explicaremos cómo fue la selección de los parámetros para este método y los resultados obtenidos con los parámetros elegidos.

La implementación de ICP utilizada es la incluida en la librería "Point Cloud Library". Esta implementación admite distintos parámetros para modificar el comportamiento del método. Los parámetros explorados son los siguientes:

1. Distancia máxima de correspondencia: Si entre dos puntos existe una distancia mayor a este valor no se van a tener en cuenta para la búsqueda de correspondencias.

2. Número de iteraciones máximo: Criterio de corte.

3. Distancia mínima entre transformaciones: Criterio de corte. Si dos transformaciones consecutivas tienen una distancia menor a este valor, el algoritmo converge.

4. Suma euclidiana mínima: Criterio de corte. Es diferencia euclideana mínima permitido entre dos pasos del algoritmo.

Además una vez que converge ICP la librería facilita el valor de la suma del cuadrado de las distancias de la nube de puntos inicial a la nube de puntos destino como medida de que tan buena es la alineación obtenida. Este valor se utilizó como umbral para decidir si la respuesta encontrada se consideraba correcta o no, por lo que también se exploró como el resto de los parámetros. A modo de refinar aún más el resultado, una vez hallada una buena alineación se procedia a tomar los puntos de la escena que suponían ser los del objeto que se estaba buscando. La explicación de cómo se realizó este filtrado se puede ver en

mbianchi: EXPLICACION USO DE KDTREE. EXPLICAR EL AUTOAJUSTE DEL LEAF PARA KDTREE

Objeto	% promedio de overlap	overlap STD	% veces seguido	Falsos Positivos	Falsos Negativos
Taza	49.12	18.74	91.67	0	2^2
Gorra	47.65	17.53	93.50	1	0
Bowl	38.24	23.50	93.94	4	0

Tab. 4.5: Resultados del tracking D utilizando como detección los valores sacados de la base de datos más una etapa de refinamiento de estos datos usando AP e ICP.

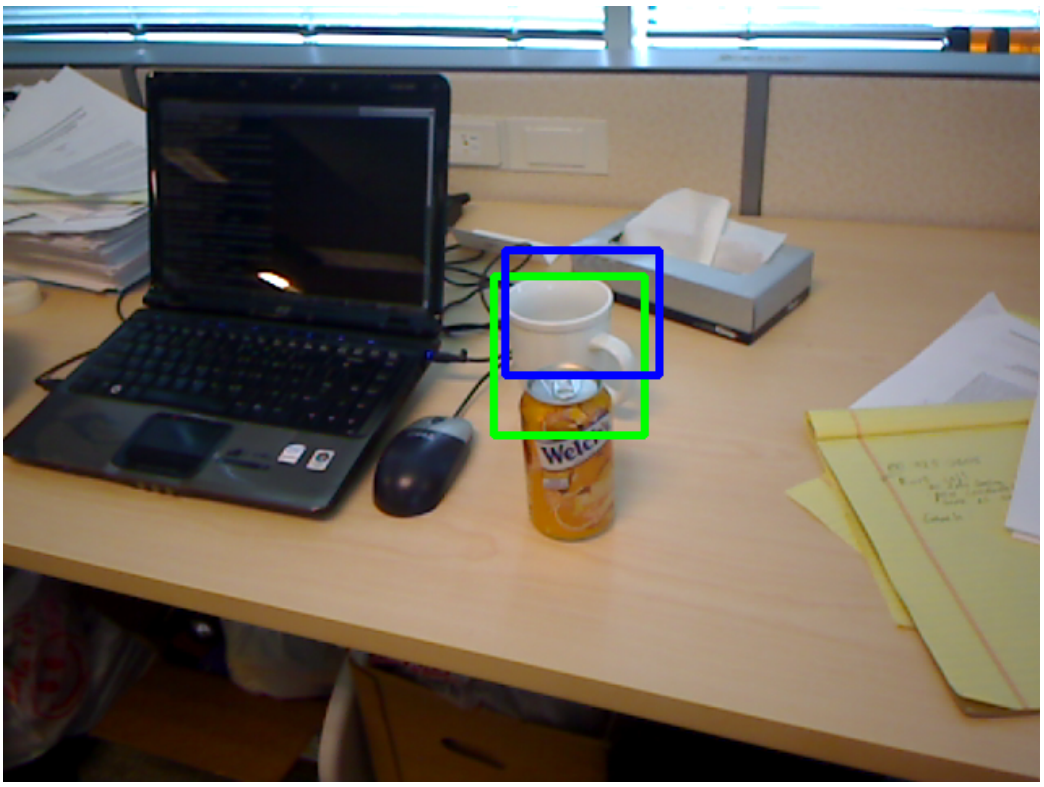
. Una vez filtrados los puntos de la escena se comparaba esta cantidad de puntos con la cantidad de puntos que tenía el modelo del objeto. Si esa cantidad superaba un porcentaje de puntos del modelo se consideraba que el objeto habia sido encontrado. Este porcentaje también fue uno de los parámetros que se exploró.

4.1.4. Evaluación del tracking en D

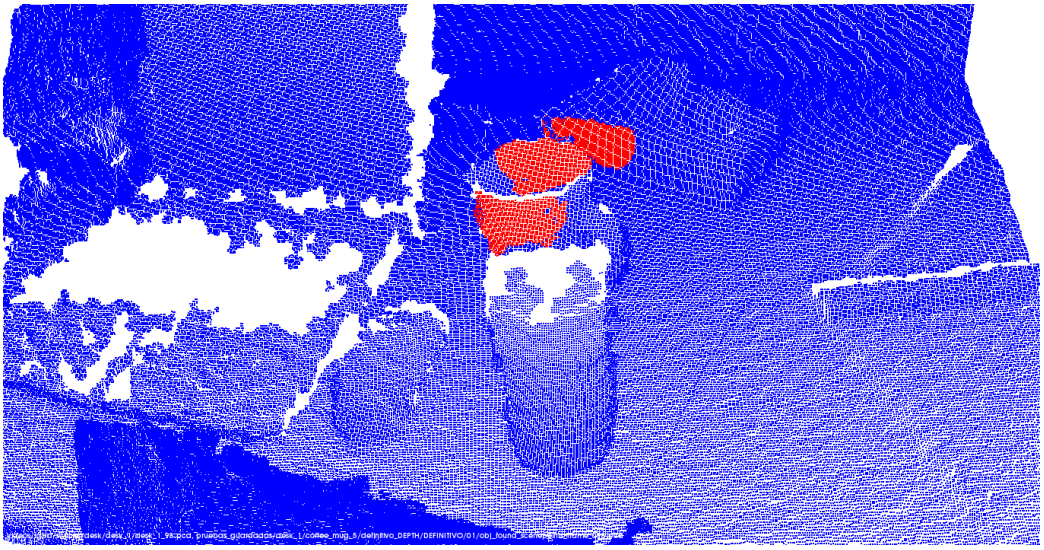
En esta sección se muestran los resultados de ICP para las imágenes D con los valores de los parámetros fijados. Para todas las pruebas se eligieron los mismos tres objetos que para el caso de la evaluación del tracking RGB.

En la tabla 4.5 se puede ver que el tracking se comporta de manera más regular que el algoritmo de RGB. Los tres ejemplos tienen una alta taza de seguimiento, en particular, todas por encima del 90 %. En los casos de la taza y la gorra además se tiene un buen balance entre el promedio de solapamiento del área encontrada y el área reportada por el ground truth y el desvío estándar.

Un problema que encontramos es que en nuestra solución no hacemos un ajuste del tamaño del área reportada si el objeto está ocluído. En la figura 4.3 podemos ver como el algoritmo detecta de manera exitosa a la taza pero que solo reporta el área de la taza que está visible y no ajusta el tamaño al del modelo del objeto que se obtuvo en la etapa de entrenamiento. La subfigura



(a) Seguimiento reportado por el tracking en D y proyectado en RGB.



(b) Visualización de la nube de puntos reportada por el tracking en D

Fig. 4.3: Muestra de un resultado de la detección en D. Las áreas se solapan en un 47 %.

4.3a muestra en color azul el recuadro reportado por nuestro algoritmo de seguimiento en donde se encuentra la taza. El recuadro verde corresponde al área que indica el ground truth como correcta. Se puede observar que el recuadro verde incluye además de la parte visible de la taza una porción importante de la lata que está delante de ella y que si se tiene en cuenta el tamaño de la taza este área está incluyendo a la parte de la taza que se encuentra ocluida. A pesar de no ser algo que suceda reiteradas veces en las escenas y objetos elegidos para estas pruebas, puede ser un punto de mejora para nuestro algoritmo.

Otra decisión que se tomó al implementar nuestro algoritmo fue que no se utilice el modelo del objeto para refinar el seguimiento, por ejemplo, para realizar el filtrado de puntos de la escena que corresponden al objeto que se busca. Esto es porque dependiendo de la forma del objeto, de la pose del objeto en el frame actual y de la pose que tiene el modelo del objeto no resulta sencillo decidir si es conveniente utilizar el modelo para hacer este filtrado. En cambio, decidimos utilizar la nube de puntos hallada en el frame anterior como base para filtrar los puntos en el frame actual una vez que se quiera refinar el resultado. Por este motivo es que en ocaciones el algoritmo devuelve como parte del resultado puntos que no corresponden al objeto que se busca. En la imagen 4.3b se pueden ver tres regiones de puntos de color rojo. Esas regiones son las reportadas por el algoritmo como puntos de la taza. Si las enumeramos de arriba hacia abajo, la primer región no forma parte de la taza sino que es parte de la caja de pañuelos que está detrás de la misma. Esto podría evitarse en este caso si se filtraran los puntos de la escena usando el modelo de la taza pero sólo porque la forma de la misma es simétrica si no se tiene en cuenta su asa. En el caso de la gorra por ejemplo ya no queda tan claro que sirva filtrar usando el modelo porque la visera la hace asimétrica y el tamaño de la misma no permite despreciar esos puntos.

Haciendo un análisis de los frames anteriores al de las figuras 4.3 y de cómo se fue desarrollando el seguimiento notamos que el motivo por el cual se incluye parte de la caja de pañuelos como puntos de la taza es porque

en la etapa de detección falló el refinamiento por AP, ICP y el filtrado de puntos. En la figura 4.4 mostramos como se ve una detección sacada directa desde la base de datos y un refinado de la detección utilizando el modelo del objeto y alineándolo usando AP e ICP y filtrando los puntos de la escena.

mbianchi: Ver frame 52,53,54,55 y decidir si conviene definir que hay deteccion si y solo si AP, ICP y el filtrado en la escena son exitosos porque cuando no lo son la nube de puntos reportada como detectada es enorme y no distingue bien cual es el objeto...

En las figuras 4.5 se muestran para cada escena y objeto como se comportó el tracking frame a frame. El análisis es el mismo que se explicó para la figura 4.2, aunque en estos gráficos aparecen algunas referencias nuevas. En la subfigura 4.5a se observan dos puntos de color naranja con valor 0 %. Estos valores corresponden a los frames 32 y 52. Estos puntos indican que el algoritmo de tracking reportó no encontrar al objeto cuando en realidad el objeto está en la imagen según los datos del ground truth. Estos puntos se corresponden con los dos falsos negativos reportados en la tabla 4.5 para la taza. En la figura 4.5b se puede identificar un punto rojo correspondiente al frame 66 con un valor del 0 %. Lo que indica este color es que el algoritmo de tracking reportó haber encontrado al objeto pero el ground truth indica que el objeto no se encuentra en ese frame, es decir, es un falso positivo. Por último, en la figura 4.5c existen múltiples puntos de color negro. Estos puntos indican que en las múltiples corridas del algoritmo para esta escena y con estos valores de parámetros hubo distintos resultados para el mismo frame. Entre los frames 72 y 108 el ground truth indica que el bowl no se encuentra en la escena. Los puntos negros en el gráfico 4.5c entre dichos frames con valor 0 % indican que alguna de las corridas el algoritmo reportó haber encontrado al objeto, es decir que hubo falsos positivos y en otras reportó correctamente que el objeto no se hallaba en esos frames. Luego, entre los frames 109 y 122 además de haber puntos negros con valor 0 % existen algunos con valores mayormente cercanos al 15 %. Eso significa que en alguna de las corridas el algoritmo reportó haber encontrado al objeto y en otras no lo encontró en

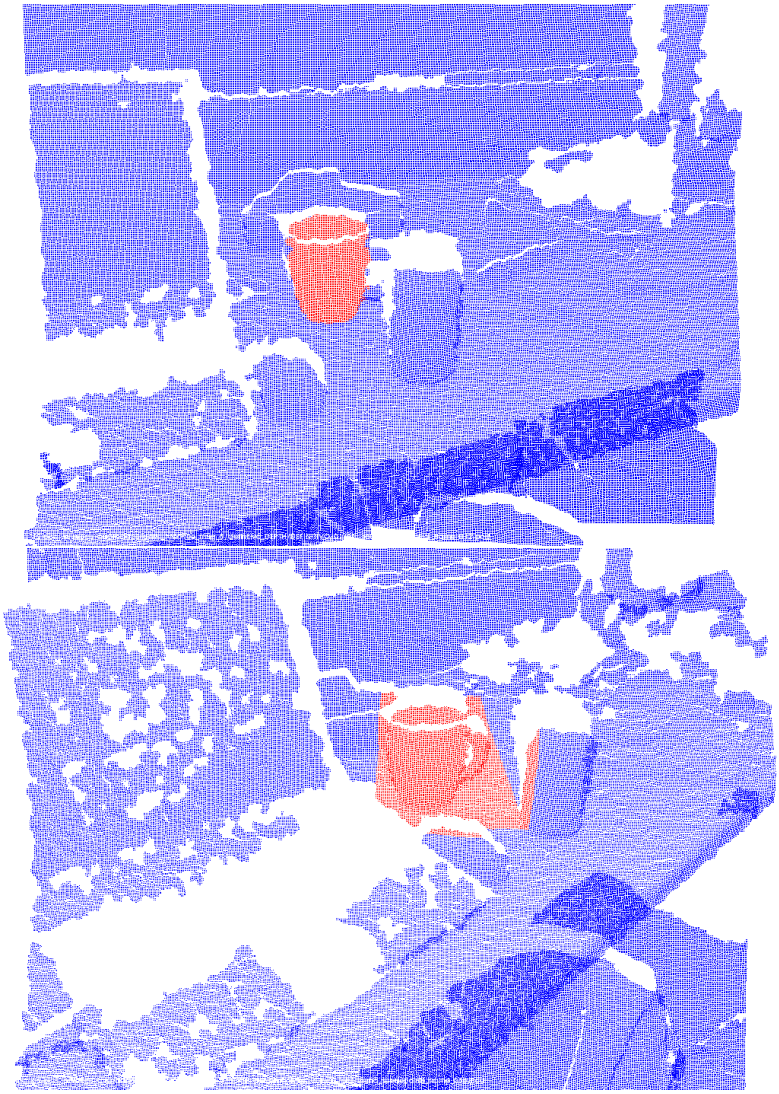
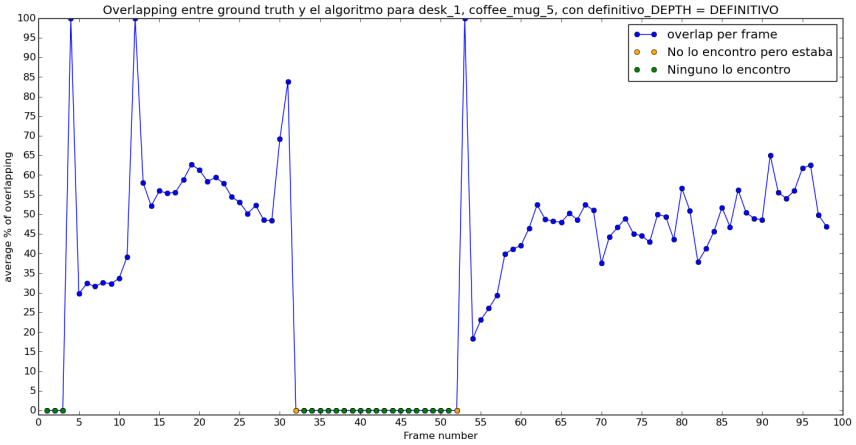


Fig. 4.4: PONER ALGO ACA

cuyo caso fueron falsos negativos. Lo que buscamos con este análisis es reducir la cantidad de puntos negros al mínimo ya que es un indicador de que tan robusto es el algoritmo.

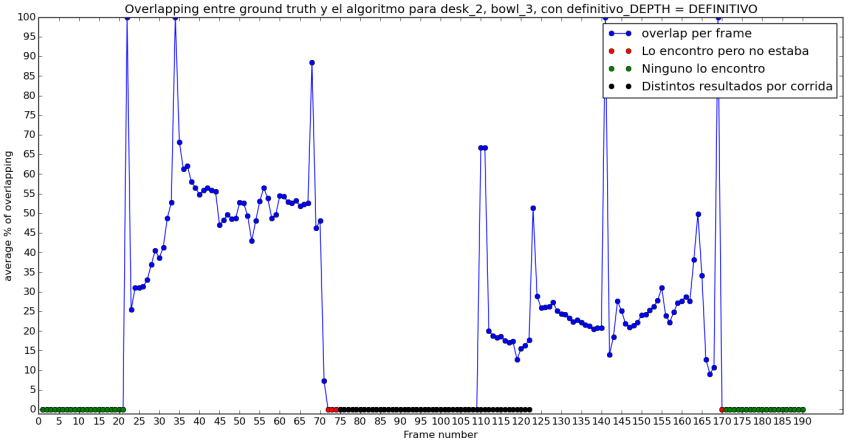
4.1.5. Evaluación del tracking en RGB-D

4.2. Discusión



(a) Seguimiento frame a frame para la taza

(b) Seguimiento frame a frame para la gorra



(c) Seguimiento frame a frame para el bowl

Fig. 4.5:

mbianchi: Descripcion

5. CONCLUSIONES

Bibliografía

[BKK+13] A.G. Buch, D. Kraft, J.-K. Kamarainen, H.G. Petersen, and N. Kruger. Pose estimation using local structure-specific shape and appearance context. In *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, pages 2080–2087, May 2013.

[BM92] P.J. Besl and Neil D. McKay. A method for registration of 3-d shapes. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 14(2):239–256, 1992.

[Bru09] Roberto Brunelli. *Template matching techniques in computer vision: theory and practice*. Wiley. com, 2009.

[DC99] Tom Drummond and Roberto Cipolla. Real-time tracking of complex structures with on-line camera calibration. In *BMVC*, pages 1–10. Citeseer, 1999.

[EBW04] Raúl San José Estépar, Anders Brun, and Carl-Fredrik Westin. Robust generalized total least squares iterative closest point registration. In *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2004*, pages 234–241. Springer, 2004.

[HLI+10] Stefan Hinterstoisser, Vincent Lepetit, Slobodan Ilic, Pascal Fua, and Nassir Navab. Dominant orientation templates for real-time detection of texture-less objects. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 2257–2264. IEEE, 2010.

[JU97] Simon J Julier and Jeffrey K Uhlmann. New extension of the kalman filter to nonlinear systems. In *AeroSense'97*, pages 182–193. International Society for Optics and Photonics, 1997.

[KRTA13] S. Korman, D. Reichman, G. Tsur, and S. Avidan. Fast-match: Fast affine template matching. In *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, pages 2331–2338, 2013.

[LBRF11] Kevin Lai, Liefeng Bo, Xiaofeng Ren, and Dieter Fox. A large-scale hierarchical multi-view rgb-d object dataset. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 1817–1824. IEEE, 2011.

[PLW11] Youngmin Park, Vincent Lepetit, and Woontack Woo. Texture-less object tracking with online training using an rgb-d camera. In *Mixed and Augmented Reality (ISMAR), 2011 10th IEEE International Symposium on*, pages 121–126. IEEE, 2011.

[SHT09] Aleksandr Segal, Dirk Haehnel, and Sebastian Thrun. Generalized-icp. In *Robotics: Science and Systems*, volume 2, page 4, 2009.

[WB95] Greg Welch and Gary Bishop. An introduction to the kalman filter, 1995.

[WVDM00] Eric A Wan and Rudolph Van Der Merwe. The unscented kalman filter for nonlinear estimation. In *Adaptive Systems for Signal Processing, Communications, and Control Symposium 2000. AS-SPCC. The IEEE 2000*, pages 153–158. IEEE, 2000.

[Zha94] Zhengyou Zhang. Iterative point matching for registration of free-form curves and surfaces. *International Journal of Computer Vision*, 13(2):119–152, 1994.