Calibración Visual

Román Velásquez Reyes

October 5, 2024

Abstract

Este documento describe el proceso de calibración visual para su uso con aplicaciones de visión computacional, principalmente para su enfoque e implementación en ORB-SLAM3.

1 Teoría de Calibración en ORB-SLAM3

La calibración de cámaras no solo implica ajustar los parámetros intrínsecos de la cámara, sino también comprender cómo se relaciona la cámara con el mundo real y otros sensores. Aquí es donde entran en juego los parámetros extrínsecos y los sistemas de referencia, que son especialmente relevantes en configuraciones avanzadas, el caso más complejo es la configuración estéreo-inerciales que se muestra en la 1, donde definimos los siguientes marcos de referencia:

1.1 Sistemas de Referencia

1. **Mundo - World (W)**: Este es un sistema de referencia fijo, donde el eje zW apunta en dirección opuesta al vector de gravedad g. La orientación y traslación en este sistema las establece el sistema SLAM, y se mantienen fijas una vez inicializadas. En sistemas puramente visuales, el sistema de referencia del mundo se define en la primera posición de la cámara.

2. **Cuerpo - Body (B)**: Esta referencia está conectada al sistema IMU (Unidad de Medición Inercial), que incluye un giroscopio y un acelerómetro, y es optimizable. Suponemos que ambos sensores comparten el mismo sistema de referencia. La pose del cuerpo TWB y la velocidad vB expresada en el sistema de referencia W son las variables que se optimizan en el proceso.

3. **Cámaras (C1 y C2):** Las cámaras coinciden con sus centros ópticos. En este sistema, el eje zC apunta hacia adelante a lo largo del eje óptico, yC apunta hacia abajo, y xC hacia la derecha, alineados con las direcciones de las imágenes u y v. Este sistema de referencia es esencial para calibraciones de cámaras estéreo, donde se necesita conocer la relación entre las dos cámaras.

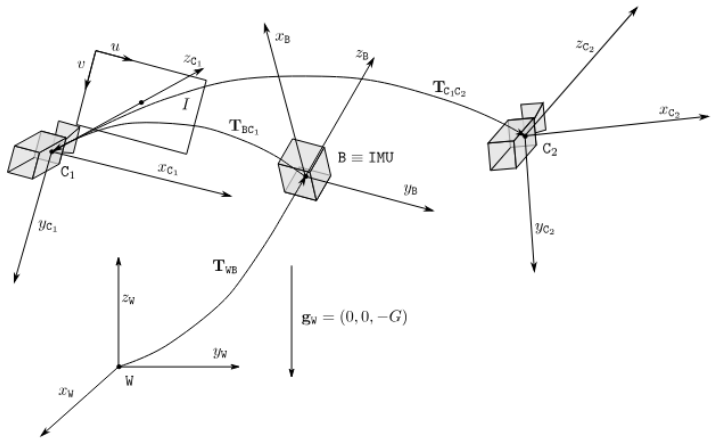

Figure 1: Sistemas de referencia definidos para el estéreo-inercial ORB-SLAM3

1.2 Parámetros Extrínsecos

Los parámetros extrínsecos son las transformaciones (rotación y traslación) que permiten pasar de un sistema de referencia a otro, como el sistema de referencia de la cámara al sistema de referencia del mundo real. Estos parámetros son cruciales para aplicaciones como ORB-SLAM3, que requieren conocer cómo se posiciona la cámara en el espacio 3D:

$$T_{WC_1} = T_{WB} T_{BC_1}$$

$$T_{WC_2} = T_{WB} T_{BC_1} T_{C_1C_2}$$

Donde los parámetros extrínsecos que debe proporcionar el archivo de calibración son:

1. Estéreo-inercial: $T_{BC_1} \quad y \quad T_{BC_1}$
2. Monocular-inercial: T_{BC_1}
3. Estéreo: solo $T_{C_1C_2}$, ORB-SLAM3 estima la pose de la cámara izquierda (B = C_1)
4. Monocular: no necesita parámetros, ORB-SLAM3 estima la pose de la cámara.

1.3 Parámetros Intrínsecos

Se refieren a las propiedades internas de la cámara, como la distancia focal (fx,fy) y el centro óptico (cx,cy), que determinan cómo la cámara proyecta los puntos del mundo real en la imagen.

La distancia focal y los centros ópticos se pueden utilizar para crear una matriz de cámara, que se utiliza para eliminar la distorsión debida a las lentes de una cámara específica. La distorsión hace que las líneas rectas parezcan curvas y se hace mayor cuanto más alejados estén los puntos del centro de la imagen. Aunque la distorsión puede ser irregular o seguir muchos patrones, podemos clasificarlas de la siguiente forma:

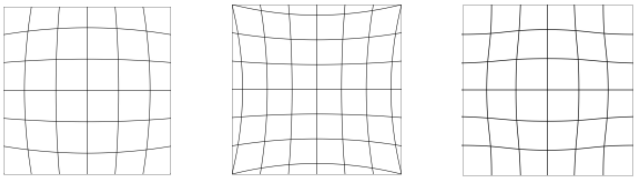

Figure 2: Ejemplos de distorsiones radiales: barril, pincushion, bigote

La distorsión radial se puede representar de la siguiente manera:

$x_{distorsionada} = x(1 + k_1r^2 + k_2r^4 + k_3r^6)$

$y_{distorsionada} = y(1 + k_1r^2 + k_2r^4 + k_3r^6)$

La distorsión tangencial ocurre porque la lente que toma la imagen no está alineada perfectamente en paralelo al plano de la imagen. Por lo tanto, algunas áreas de la imagen pueden verse más cercanas de lo esperado. La cantidad de distorsión tangencial se puede representar de la siguiente manera:

$x_{distorsionada} = x + [2p_1xy + p_2(r^2 + 2x^2])$

$y_{distorsionada} = y + [p_1(r^2 + 2y^2) + 2p_2xy])$

En resumen, necesitamos encontrar cinco parámetros, conocidos como coeficientes de distorsión dados por:

Coeficientes de distorsión = $(k_1, k_2, p_1, p_2, k_3)$

Además de esto, necesitamos otra información, como la matriz de la cámara, que es exclusiva de una cámara específica, por lo que una vez calculada, se puede reutilizar en otras imágenes tomadas por la misma cámara. Se expresa como una matriz de 3x3:

$$Matriz \quad camara = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \tag{1}$$

Dependiendo de la configuración de la cámara, necesitaremos proporcionar distintos parámetros de calibración.

1.3.1 Modelos de Cámara en ORB-SLAM3

ORB-SLAM3 soporta diferentes tipos de modelos de cámara:

1. **Cámara Pinhole**: Se utilizan cámaras estenopeicas. Este modelo es ideal para cámaras que no tienen distorsiones significativas en las imágenes.
   Los parámetros intrínsecos que se deben proporcionar para una cámara Pinhole son:

   - **Distancia focal (fx,fy)**: Indican la distancia focal en píxeles en los ejes x e y, respectivamente.
   - **Punto central (cx,cy)**: Indican el punto en la imagen donde se cruzan los ejes ópticos, también en píxeles.

   El modelo Pinhole incluye un modelo de distorsión radial-tangencial para corregir las distorsiones en la imagen. Este modelo utiliza:

   - **Coeficientes de distorsión radial (k1,k2,k3)**: Estos parámetros se encargan de corregir la distorsión que hace que las líneas rectas aparezcan curvas, un efecto común en cámaras con lentes.
   - **Coeficientes de distorsión tangencial (p1,p2)**: Ajustan cualquier distorsión provocada por la inclinación del lente respecto al plano de la imagen.

2. **Cámara KannalaBrandt8**: Es adecuado para cámaras con lentes de gran angular y ojo de pez. Este tipo de lentes suelen presentar distorsiones significativas, por lo que se requiere un modelo de distorsión especializado para corregirlas.
   Los parámetros intrínsecos que se deben proporcionar para una cámara Kannala-Brandt8 son:

   - **Distancia focal (fx,fy)**
   - **Punto central (cx,cy)**
   - **Coeficientes de distorsión radial (k1,k2,k3)**

3. **Cámara Rectificada (Rectified)**: Se utiliza para cámaras o datasets que proporcionan imágenes estéreo rectificadas. En este caso, las imágenes ya han sido procesadas para corregir las distorsiones y alinearlas en una geometría estéreo. Para este caso el archivo solo necesita Distancia focal (fx,fy) y el punto central (cx,cy).

La elección del modelo depende del tipo de cámara utilizado y afecta cómo se almacenan los parámetros en un archivo de calibración nombrado ".yaml".

1.4 Tablero

La calibración es un proceso que nos permite encontrar los parámetros extrínsecos e intrínsecos de una cámara. Estos parámetros nos ayudan a corregir distorsiones en las imágenes y a determinar cómo los puntos 3D del mundo real se proyectan en 2D en una imagen. Se pueden obtener utilizando un software de calibración como OpenCV, Matlab o ROS, como se explica en las siguientes secciones:

Para encontrar estos parámetros, debemos proporcionar algunas imágenes de muestra de un patron definido, como un tablero de ajedrez

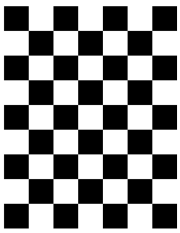

Figure 3: Tablero

Identificamos algunos puntos específicos cuyas posiciones relativas ya conocemos, como las esquinas de los cuadrados del tablero de ajedrez. Conocemos las coordenadas de estos puntos en el espacio del mundo real (midiendo la distancia entre los cuadros) y en la imagen, lo que nos permite resolver los coeficientes de distorsión. Se recomienda trabajar con al menos de 10 a 20 fotografias o patrones de prueba para obtener mejores resultados.

Los datos de entrada importantes necesarios para la calibración de la cámara son el conjunto de puntos 3D del mundo real y las coordenadas 2D correspondientes de estos puntos en la imagen.

Puntos 2D: Podemos verificar si los puntos en 2D son corrector, utilizando la imagen post-procesada, es decir: estos puntos se deben encontrar como intersección de dos cuadrados negros en el tablero de Ajedrez.

Puntos 3D: Esas imágenes se toman desde una cámara estática y los tableros de ajedrez se colocan en diferentes ubicaciones y orientaciones. Por lo tanto, necesitamos saber los valores (X, Y, Z). Para simplificar y evitar mayor complejidad, podemos mantener el tablero de ajedrez estacionario en el plano XY

(por lo que Z = 0 siempre).

Esta consideración nos ayuda a encontrar solo valores X, Y. Ahora, para los valores X, Y, simplemente podemos pasar los puntos como (0, 0), (1, 0), (2, 0), ... que denota la ubicación de los puntos. En este caso, los resultados que obtenemos estarán en la escala del tamaño del cuadrado del tablero de ajedrez, la mayoria de los métodos solicitan el tamañano de los cuadrados en m, cm o mm.

Los puntos 3D se denominan puntos de objeto y los puntos de imagen 2D se denominan puntos de imagen.

2 Archivo.yaml

El archivo .yaml contiene toda la información de los parámetros con los que trabaja la cámara. El autor original resume todos los parametros que ORB-SLAM3 requiere en cualquiera de sus etapas, incluyendo parámetros de calibración intrínsecos y extrínsecos, parámetros de extracción de ORB y configuraciones de visualización.
Todos estos parámetros de configuración, los requiere ORB-SLAM3 en un archivo .yaml.

2.1 Parámetros generales de calibración

Parámetros generales de calibración			
Parámetro	Valor	Obligatorio	Definición
File.version	1.0	Si	Especifica que el uso del nuevo formato de archivo de calibración.
Camera.type	string	Si	Tipo de cámara que se está utilizando, como: PinHole, KannalaBrandt8 y Rectificadas.
Camera.height, Camera.width	int	Si	Tamaño del alto y ancho de la imagen de entrada
Camera.newHeight, Camera.newWidth	int	Opcional	Cambia el tamaño de las imágenes de entrada a la nueva resolución especificada.
Camera.fps	int	Si	Fotogramas por segundo de la secuencia de vídeo.
Camera.RGB	int	Si	Especifica si las imágenes son: BGR (0) o RGB(1)
System.thFarPoints	float	Opcional	Especifica la profundidad máxima en metros permitida para un punto. Los puntos más alejados se ignoran.

2.2 Parámetros intrínsecos

Parámetros intrínsecos de la cámara			
Parámetro	Valor	Obligatorio	Definición
Definimos la Camera1 como la cámara monocular (en SLAM monocular) o la cámara izquierda (en SLAM estéreo). Sus parámetros intrínsecos corresponden a:			
Camera1.fx, Camera1.fy, Camera1.cx, Camera1.cy	float	Si	corresponde a los parámetros de calibración intrínsecos de la cámara 1.
Si Camera.type está configurado como PinHole, debe especificar:			
Camera1.k1, Camera1.k2	float	Si	Corresponde a los coeficientes de distorsión radial.
Camera1.p1, Camera1.p2	float	Si	Corresponde a los coeficientes de distorsión tangencial.
Camera1.k3	float	Opcional	Un tercer parámetro de distorsión radial es opcional. Puede especificarlo con este parámetro

2.3 Parámetros ORB

Los siguientes parámetros están relacionados con la extracción de características ORB:

Parámetros ORB			
Parámetro	Valor	Obligatorio	Definición
ORBextractor.nFeatures	int	Si	Número de características a extraer por imagen.
ORBextractor.scaleFactor	float	Si	Factor de escala entre niveles en la pirámide de la imagen.
ORBextractor.nLevels	int	Si	Número de niveles en la pirámide de la imagen.
ORBextractor.iniThFAST	int	Si	Umbral inicial para detectar esquinas FAST.
ORBextractor.minThFAST	int	Si	Si no se encuentran características con el umbral inicial, el sistema intenta una segunda vez con este valor de umbral.

2.4 Parámetros Atlas

Estos parámetros definen si cargamos/guardamos un mapa desde/hacia un archivo:

Parámetros Atlas			
Parámetro	Valor	Obligatorio	Definición
System.LoadAtlasFromFile	string	Opcional	Ruta del archivo donde se encuentra el mapa a cargar.
System.SaveAtlasToFile	string	Opcional	Archivo de destino donde guardar el mapa generado.

2.5 Parámetros del visor

Estos son algunos parámetros relacionados con la interfaz de usuario de ORB-SLAM3:

Parámetros Atlas			
Parámetro	Valor	Obligatorio	Definición
Viewer.KeyFrameSize	float	Si	Tamaño en el que se dibujan los KeyFrames en el visor de mapas.
Viewer.KeyFrameLineWidth	float	Si	Ancho de línea del dibujo del KeyFrame.
Viewer.GraphLineWidth	float	Si	Ancho de línea del gráfico de covisibilidad.
Viewer.PointSize	float	Si	Tamaño del dibujo del MapPoint.
Viewer.CameraSize	float	Si	Tamaño en el que se dibuja la cámara actual en el visor de mapas.
Viewer.CameraLineWidth	float	Si	Ancho de línea del dibujo de la cámara actual.
Viewer.ViewpointX, Viewer.ViewpointY, Viewer.ViewpointZ, Viewer.ViewpointF	float	Si	Punto de vista inicial del visor de mapas.
Viewer.imageViewScale	float	Opcional	Factor de cambio de tamaño para la visualización de imágenes (solo para visualización, no se utiliza en la secuencia de SLAM).

Con todos estos parámetros de calibración que incorpora el archivo .yaml para usarlo junto con ORB-SLAM3. Recomendable crear o actualizar el archivo de configuración .yaml para el formato TUM.

3 Calibración

Existen distintos metodos, de sistemas calibrados funcionan a partir de una tabla de funciones de transferencia de lente/cámara:

3.1 Calibración con ROS:

La métodologia se puede implementar para cualquier dispositivo que utilice camara monocular, para este ejemplo se realizara para una cámara webcam.

3.1.1 Verificar detección del dispositivo o webcam

1. Conecta la webca al puerto usb, como primera revisión, abre la terminal y ejecuta "Cheese", este es un software de ubuntu para tomar fotografias y video.

   ```
   cheese
   ```

2. Otra forma de confirmar si el dispositivo de cámara web está detectado en el sistema, usando la terminal, con el siguiente comando:

   ```
   ls -l /dev/video*
   #En la salida:
   crw-rw----+ 1 root video 81, 0 sep  3 11:28 /dev/video0
   crw-rw----+ 1 root video 81, 1 sep  3 11:28 /dev/video1
   o támbien
   ls /dev/ | grep video
   ```

```
video0 #lo nombra por default
video1 #Son 2 salidas por webcam
```

Las diecciones de los dispositivos la utilizaremos para la salida del nodo de ROS

3.1.2 Calibración como Nodo de ROS

1. Instala el paquete de ROS usb_cam con los siguientes comandos

```
sudo apt install ros-noetic-usb-cam
sudo apt install ros-noetic-perception
```

2. Dirección de la cámara en nodo, accede al nodo de ros_webcam y verifica el parametro de video_device

```
#Accede a la carpeta del nodo.
cd ~/opt/ros/noetic/share/usb_cam/launch
#Abre el archivo \usb_cam-test.launch" y el valor de value,
debe ser igual al dispositivo que detecto.
<param name="video_device" value="/dev/video0" />
```

3. Descarga e imprime el archivo Tablero.

4. Ejecutar el nodo de ros-webcam

```
terminal 1: roscore
terminal 2: roslaunch usb_cam usb_cam-test.launch
#Se abrira una ventana de la transmisión de video de la webcam
```

5. El paquete incluye un nodo para calibrar la cámara, lo inicializamos con el siguiente comando:

```
terminal 3:
rosrun camera_calibration cameracalibrator.py --size 8x6
--square 0.0245 imagen:=/usb_cam/image_raw camara:=/usb_cam
#El valor de square se refiere al tamaño de los cuadrados en metros
```

6. Al finalizar, selecciona el botón de “save” para guardar nuestros valores intrinsecos y extrinsicos.

7. Crear un archivo .yaml y acomodar los valores de la matrix como el ejemplo del archivo TUM.yaml. El siguiente video muestra el resultado: #Donde se guarda la info y sustituirla

3.2 Calibración con Matlab:

Para calibrar la cámara utilizando un tablero de ajedrez, podemos usar la herramienta de camera_calibration en las aplicaciones. Para este ejemplo utilizamos la cámara del dron DJI Tello Ryze.

1. Tomar fotografias al tablero de ajedrez.
2. Abrir la herramienta de camera_calibration.
3. Insertar las fotografias e indicar qué tipo de patrón estamos buscando, como una cuadrícula de 8x8, una cuadrícula de 5x5, etc. Para este caso utilizamos una cuadrícula de 7x6.
4. Devuelve los puntos de las esquinas y el valor de retorno, que será verdadero si se obtiene el patrón. Estas esquinas se colocarán en un orden (de izquierda a derecha, de arriba a abajo).
5. A continuación se muestra una imagen con un patrón dibujado: (imagen del tablero con lineas)
6. Podemos elegir las mejores imagenes para disminuir el error.
7. Una vez observado los resultados, podemos seleccionar la opcion de calibración y de obtener el script que devuelve la matriz de la cámara, los coeficientes de distorsión, los vectores de rotación y traslación, etc.

```
int main() {
std::cout << "Hola, mundo!" << std::endl;
return 0;
}
```