Matchmove

Multimedia Techniques & Applications

Yu-Ting Wu

(with slides borrowed from Prof. Yung-Yu Chuang)

Jurassic Park (1993)

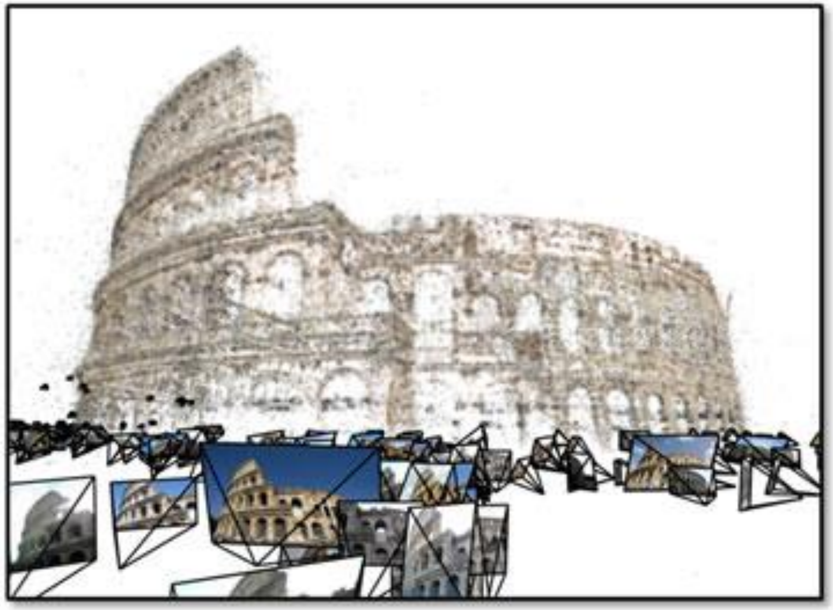
How to Composite Virtual and Real?

- In the real world, we use a **camera** to record the information of the real scene
- In a virtual world, we use a **virtual camera** to record the information of the virtual scene
- **Idea:** make the virtual camera **sync** with the real-world camera and **put the virtual objects in the right places**

How to Composite Virtual and Real? (cont.)

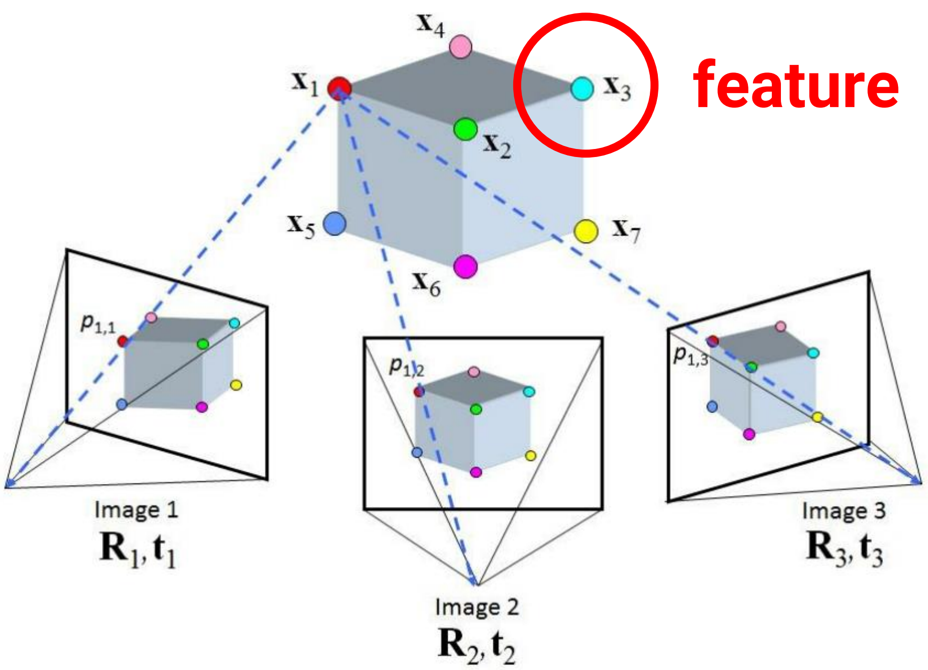
Matchmove (Structure from Motion)

- **Structure from Motion:** automatic recovery of camera motion and scene structure from two or more images
- Also called **matchmove** in film production



Matchmove (Structure from Motion)

- **Structure from Motion:** automatic recovery of camera motion and scene structure from two or more images
- Also called **matchmove** in film production



Features

- Also known as **interesting points**, **salient points**, or **keypoints**
- Points that you can easily point out their **correspondences** in **multiple images** using only **local information**

Desired Properties for Features

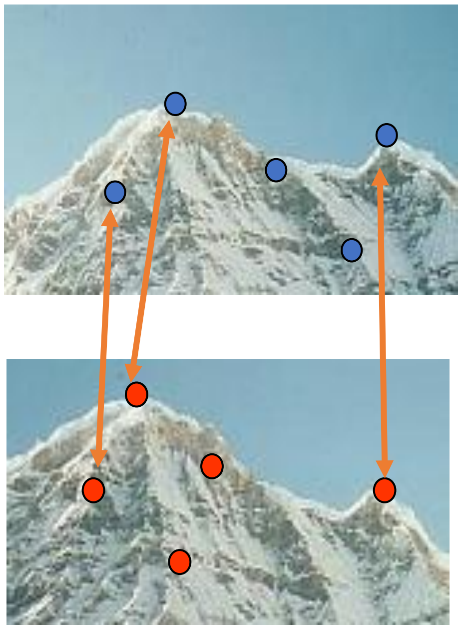
- **Distinctive**
    - A single feature can be correctly matched with high probability
- **Invariant**
    - Invariant to scale, rotation, illumination and noise for robust matching across a substantial range of distortion, viewpoint change and so on

Applications

- **Object or scene recognition**
- **Matchmove (structure from motion)**
- **Stereo**
- **Motion tracking**
- **…**

Components

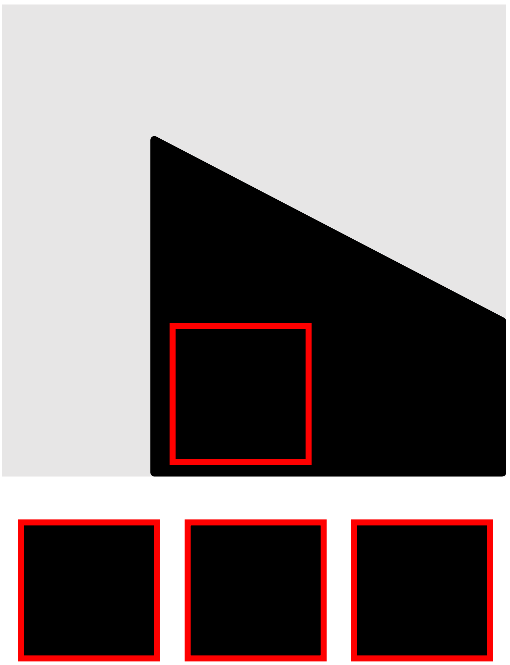
- **Feature detection** locates where they are
- **Feature description** describes what they are
- **Feature matching** decides whether two are the same one



Moravec Corner Detector

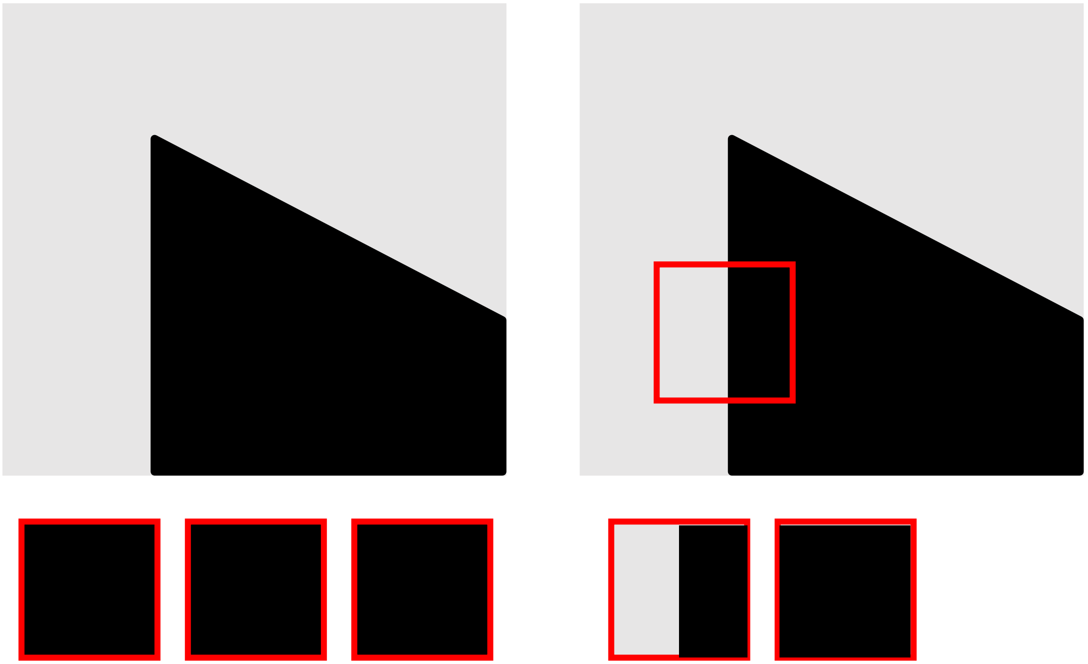
- We should easily recognize the point by looking through a **small window**
- Shifting a window in any direction should give a large change in **intensity**

Moravec Corner Detector (cont.)



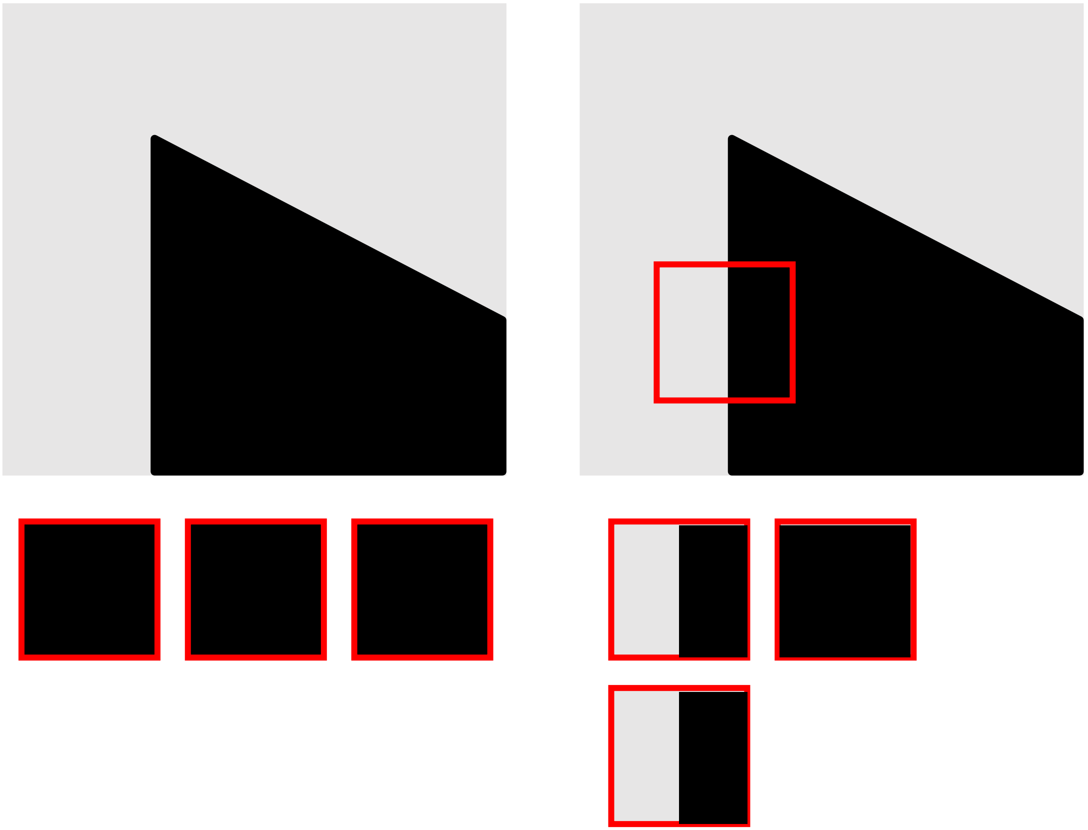
flat

Moravec Corner Detector (cont.)



flat

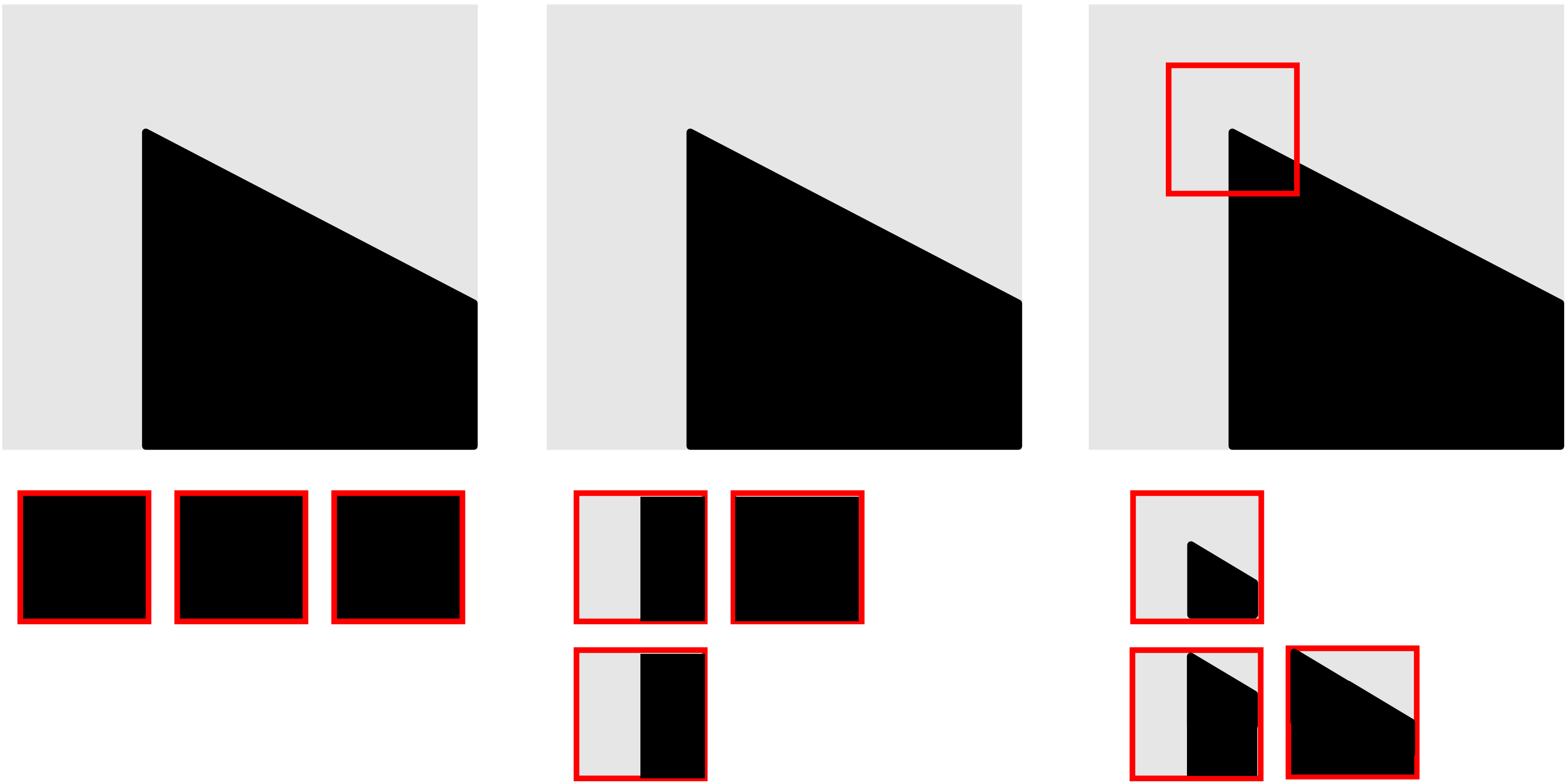
Moravec Corner Detector (cont.)



flat

edge

Moravec Corner Detector (cont.)



flat

edge

corner
isolated point

Moravec Corner Detector (cont.)

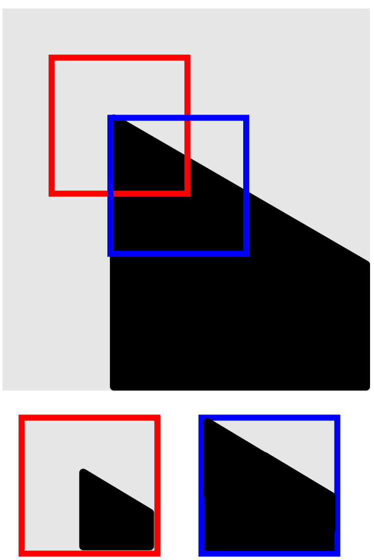
- Change of intensity for the shift [u, v]

$$E(u,v) = \sum_{x,y} w(x,y)\left[I(x+u, y+v) - I(x,y)\right]^2$$

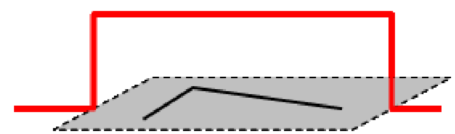
window function

shifted intensity

intensity



Window function $w(x,y) =$



1 in window, 0 outside

Four shifts: (u,v) = (1,0), (1,1), (0,1), (-1, 1)

Look for local maxima in *min{E}*

Problems of Moravec Detector

- Noisy response due to a binary window function
- Only a set of shifts at every 45 degree is considered
- Only minimum of E is taken into account

➔ Harris corner detector solves these problems

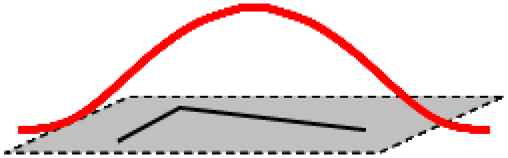
Harris Corner Detector

- Noisy response due to a binary window function

➔ **Use a Gaussian function**

$$w(x, y) = \exp\left(-\frac{(x^2 + y^2)}{2\sigma^2}\right)$$

Window function $w(x,y) =$



Gaussian

Harris Corner Detector (cont.)

- Only a set of shifts at every 45 degree is considered

➔ **Consider all small shifts by Taylor's expansion**

$$E(u,v) = \sum_{x,y} w(x,y)\left[I(x+u, y+v) - I(x,y)\right]^2$$

$$= \sum_{x,y} w(x,y)\left[I_x u + I_y v + O(u^2, v^2)\right]^2$$

$$A = \sum_{x,y} w(x,y) I_x^2(x,y)$$

$$B = \sum_{x,y} w(x,y) I_y^2(x,y)$$

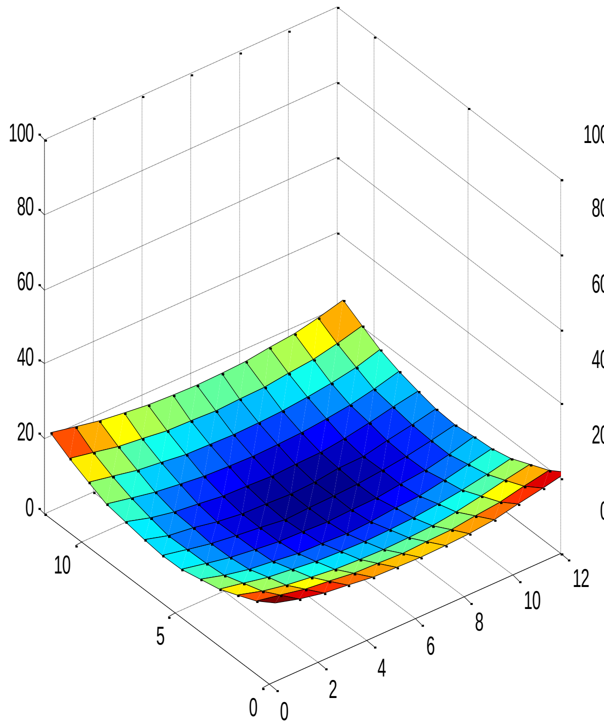
$$C = \sum_{x,y} w(x,y) I_x(x,y) I_y(x,y)$$

➡ $E(u,v) = Au^2 + 2Cuv + Bv^2$

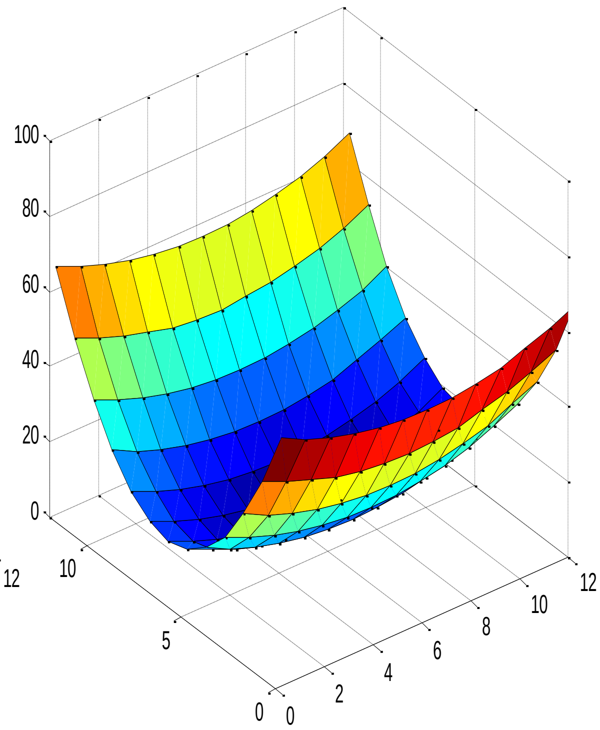
We can obtain a new measurement by investigating the shape of the error function

Harris Corner Detector (cont.)

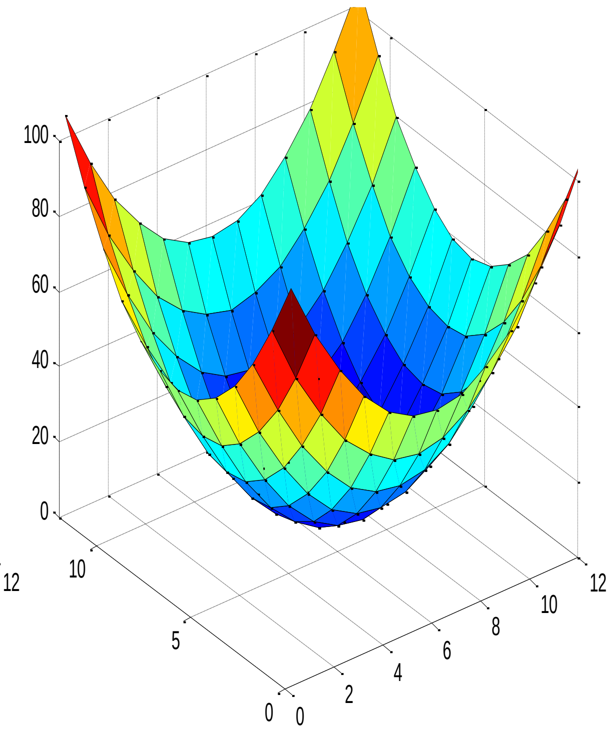
- High-level idea: what shape of the error function will we prefer for features?



flat



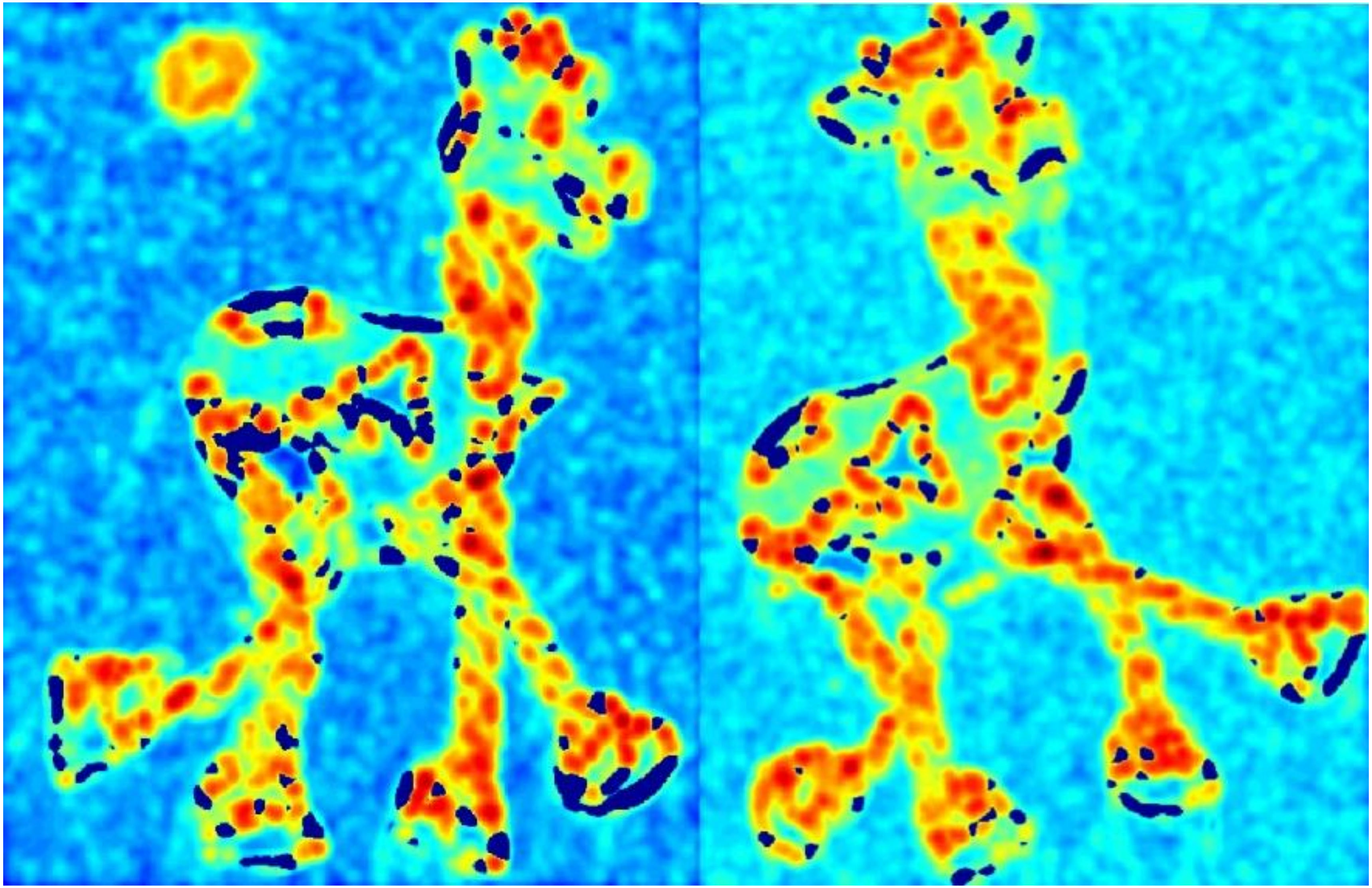
edge



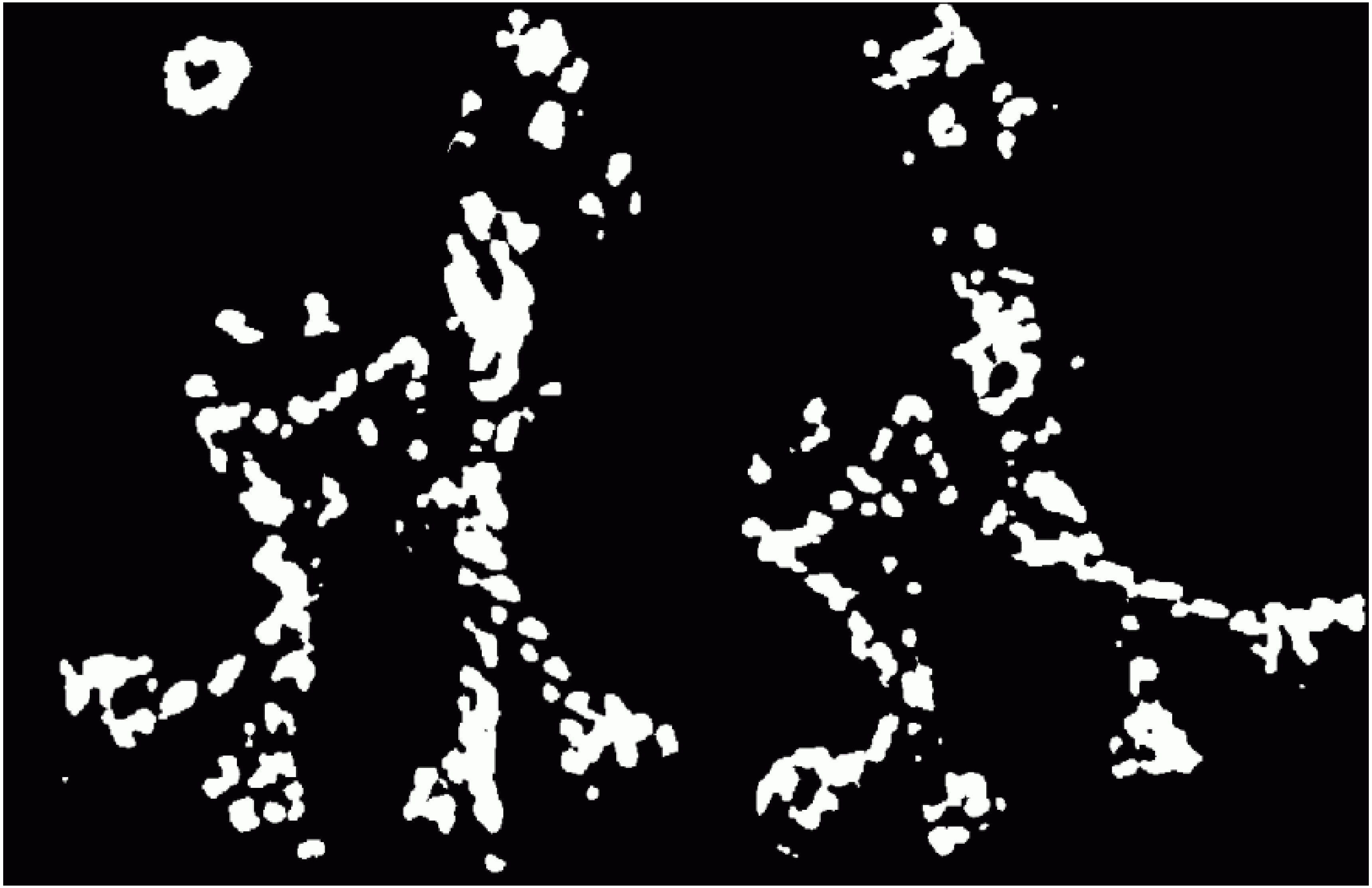
corner

Harris Corner Detector (Input)

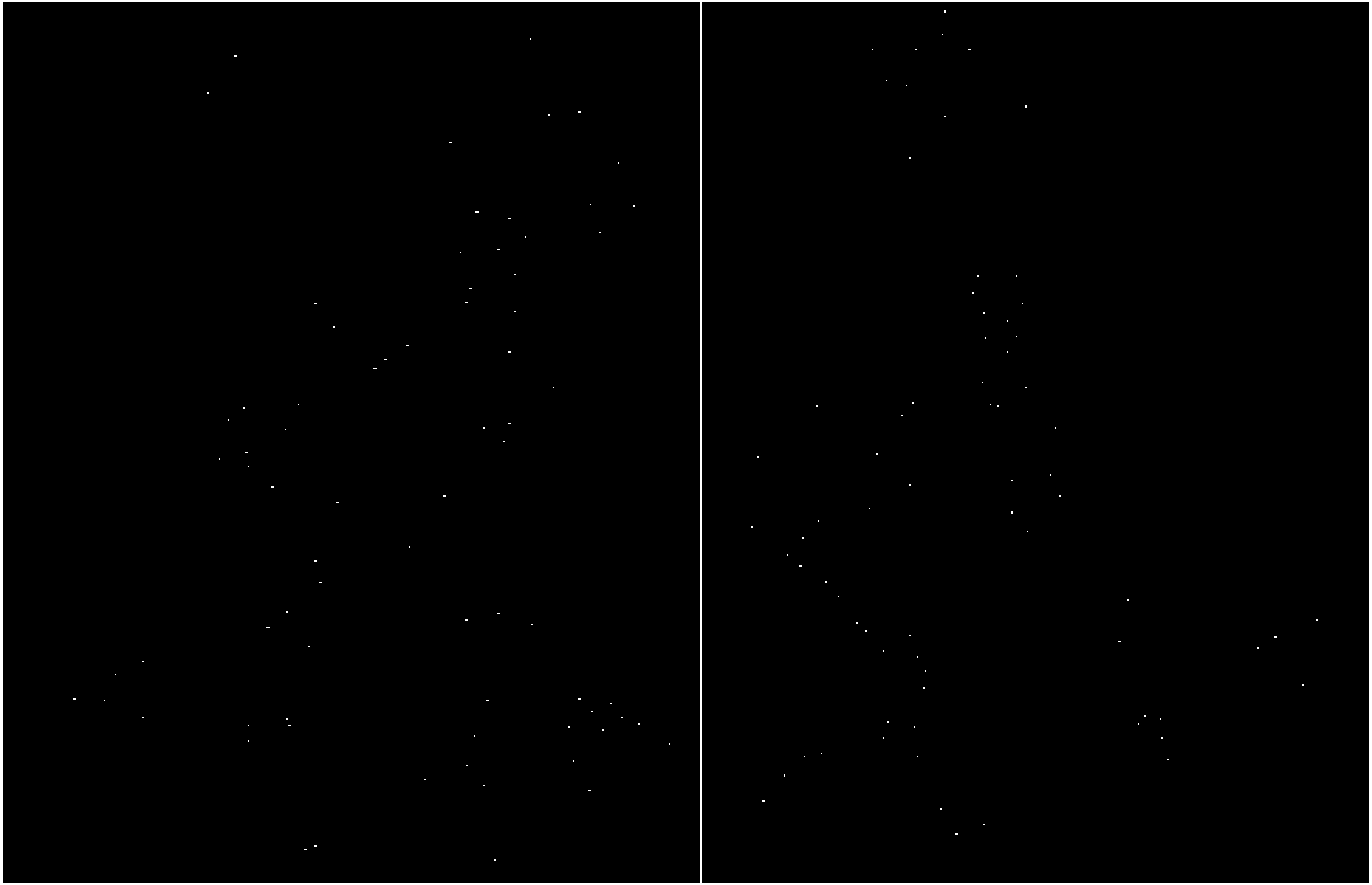
Harris Corner Detector (Response)



Harris Corner Detector (Theshold)



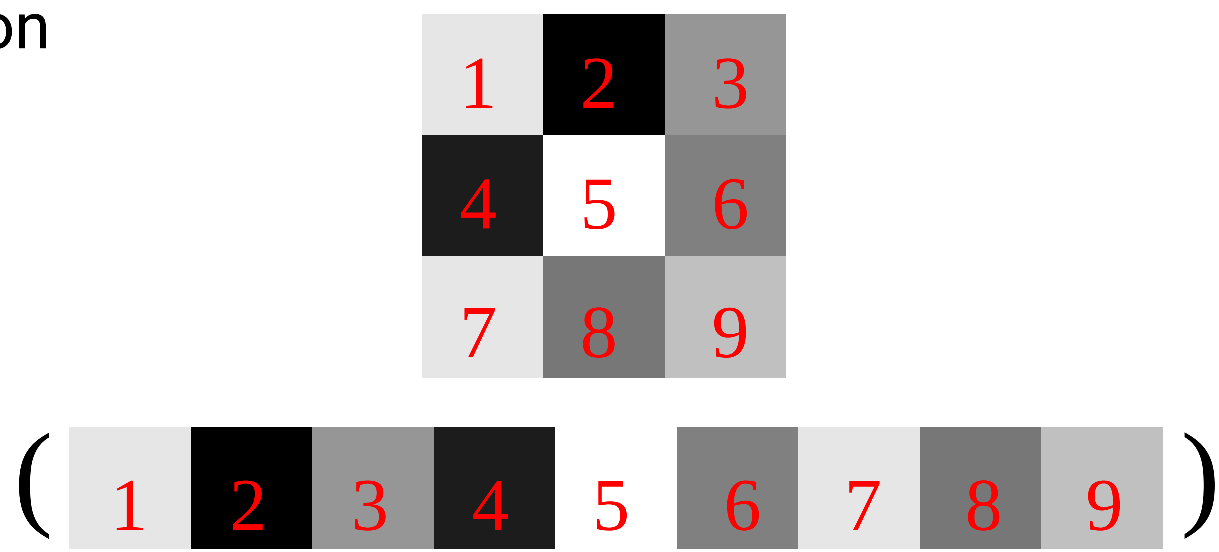
Harris Corner Detector (Local Maximum)



Harris Corner Detector (Output)

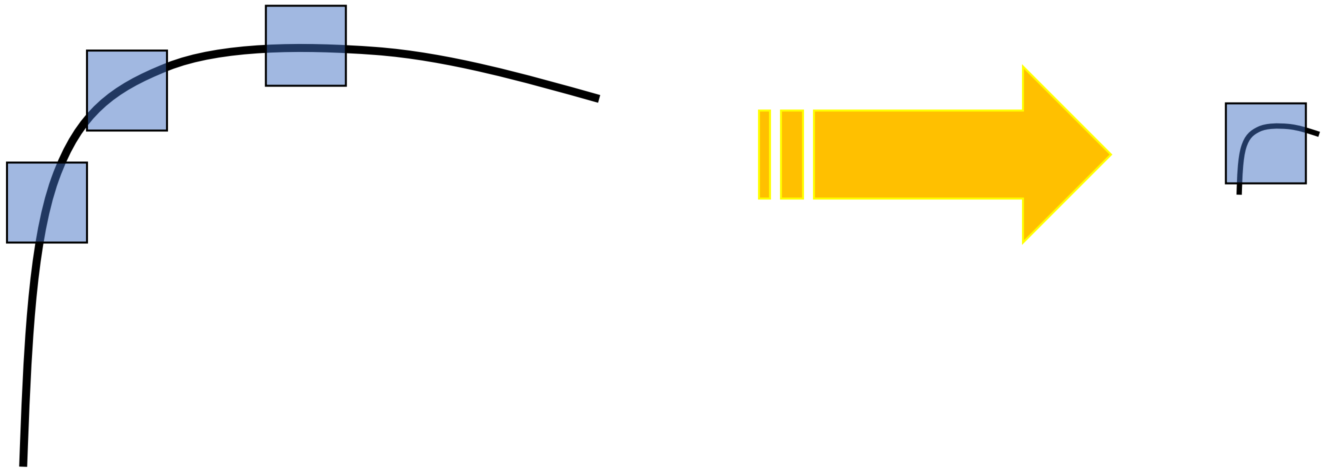
Feature Description

- Now we know where the features are
- But how to match them?
- What is the descriptor for a feature? The simplest solution is the intensities of its spatial neighbors
- This might not be robust to brightness change or small shift/rotation



Problems of Harris Detector

- Not invariant to image scale

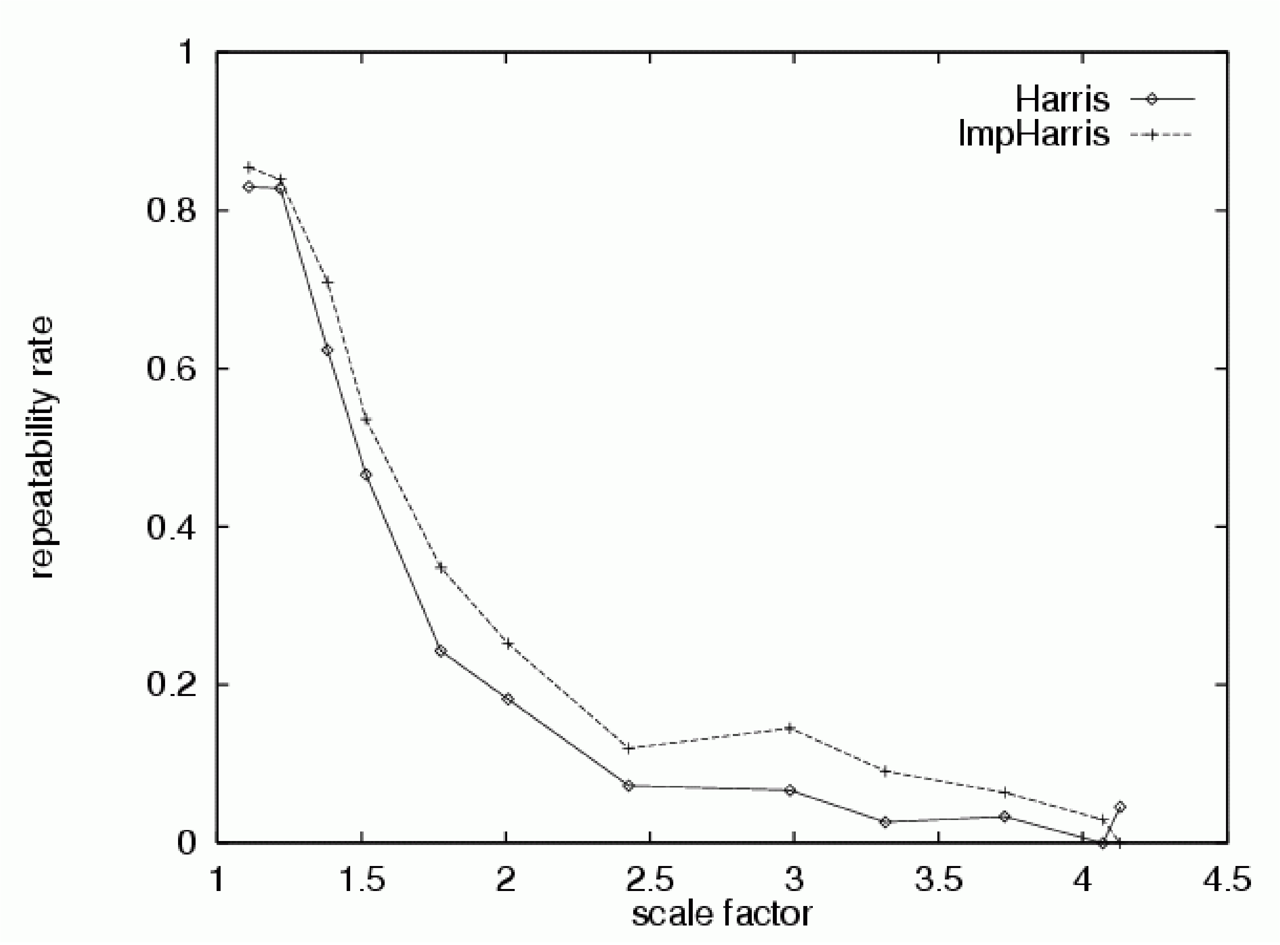


All points will be classified as edges

Corner !

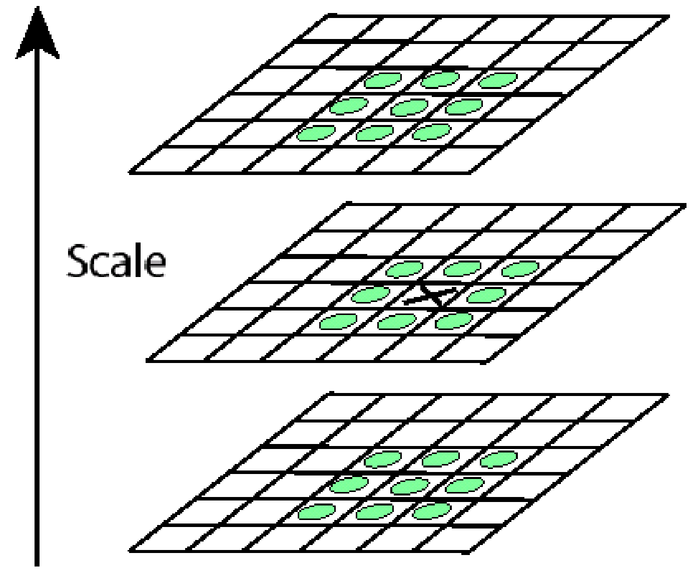
Problems of Harris Detector (cont.)

- Not invariant to image scale



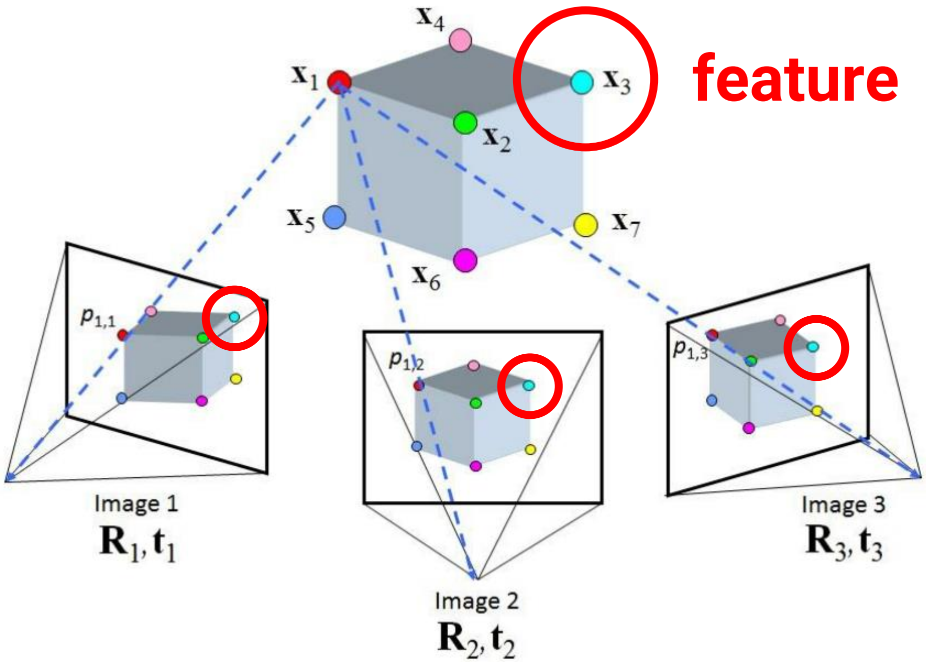
SIFT

- Stands for **Scaled Invariant Feature Transform**
- For **scale invariance**, search for stable features **across all possible scales** using a continuous function of scale, scale space.

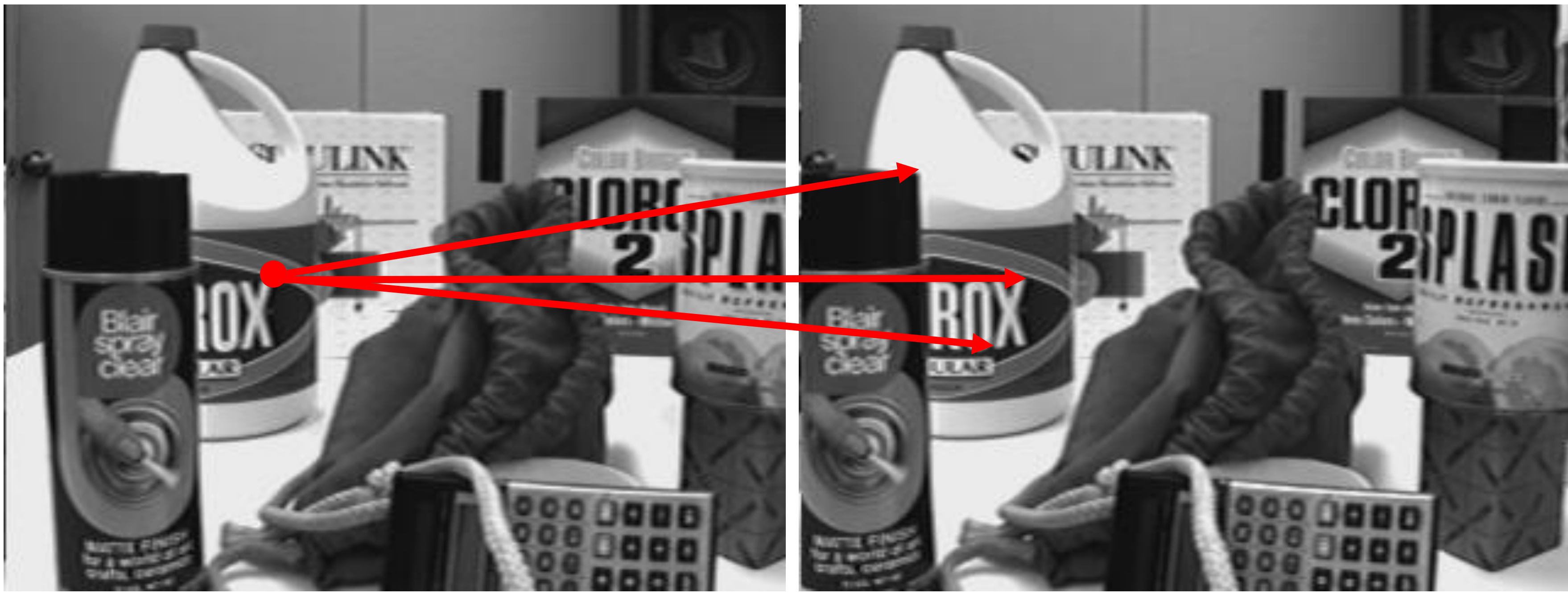


Tracking

- If we detect a feature point in one frame, how do we keep tracks of it in other frames?



Tracking (cont.)

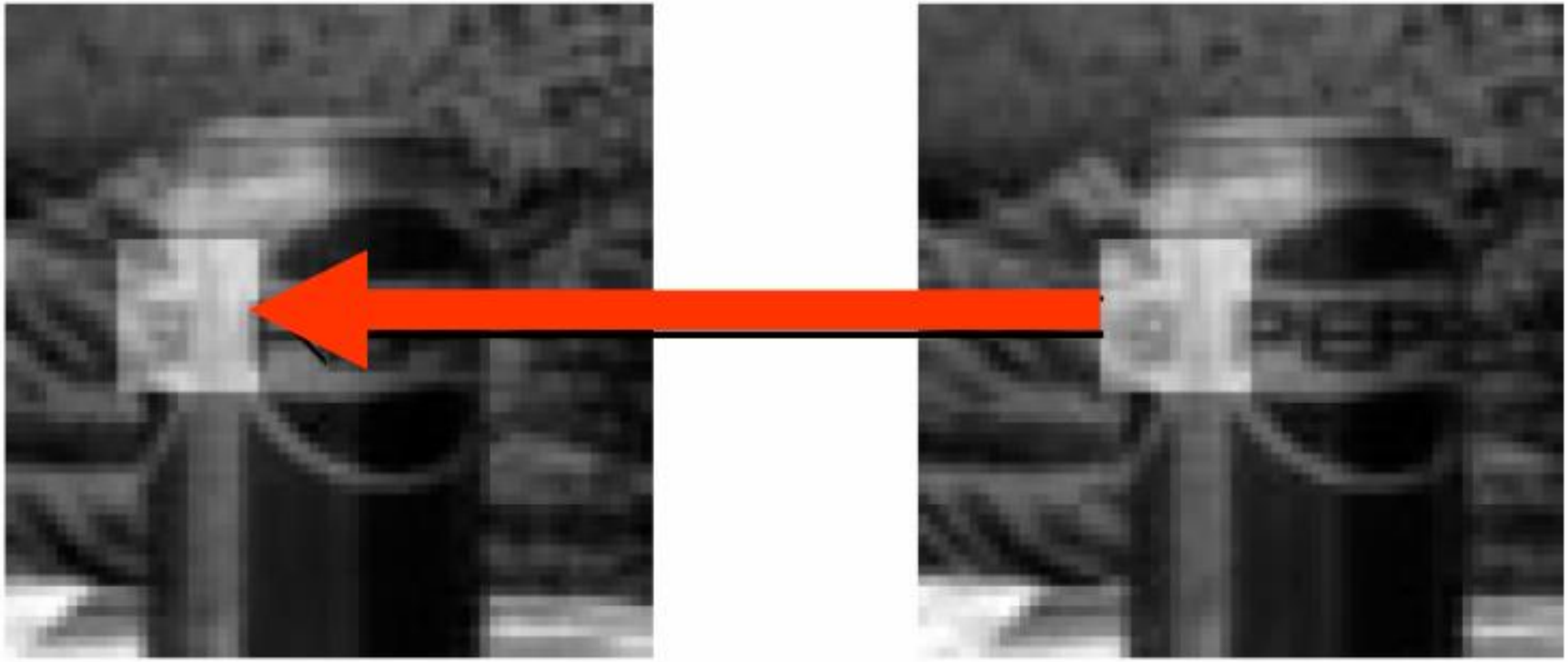


Three Assumptions of Tracking

- **Brightness consistency**
- **Spatial coherence**
- **Temporal persistence**

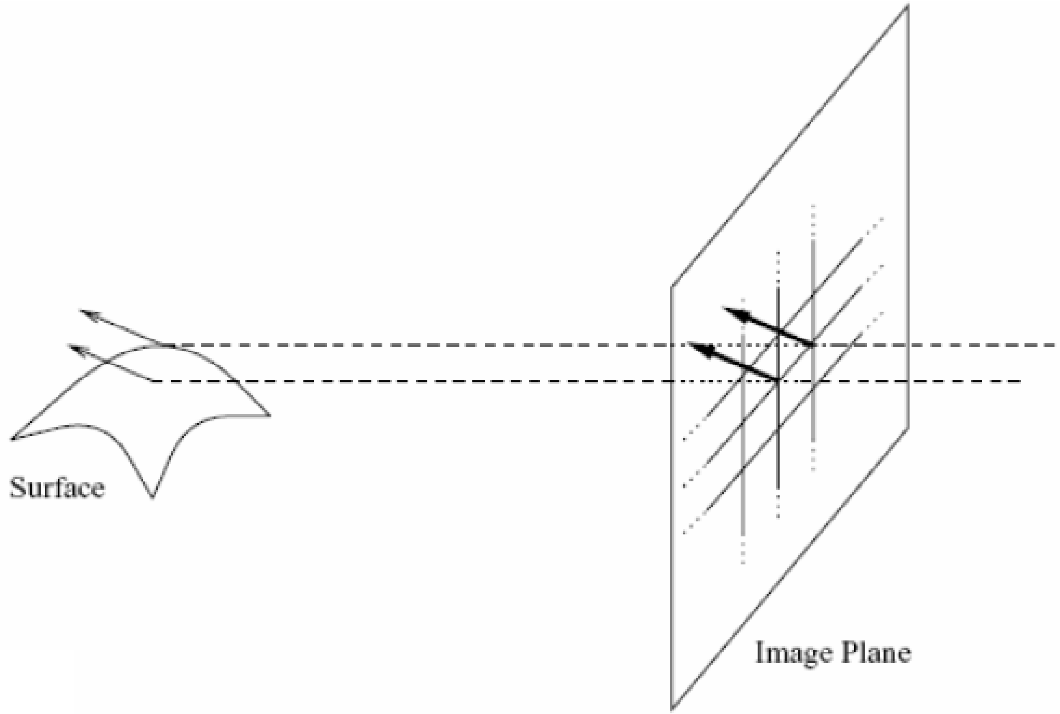
Brightness Consistency

- Image measurement (e.g. brightness) in a small region remain the same although their location may change



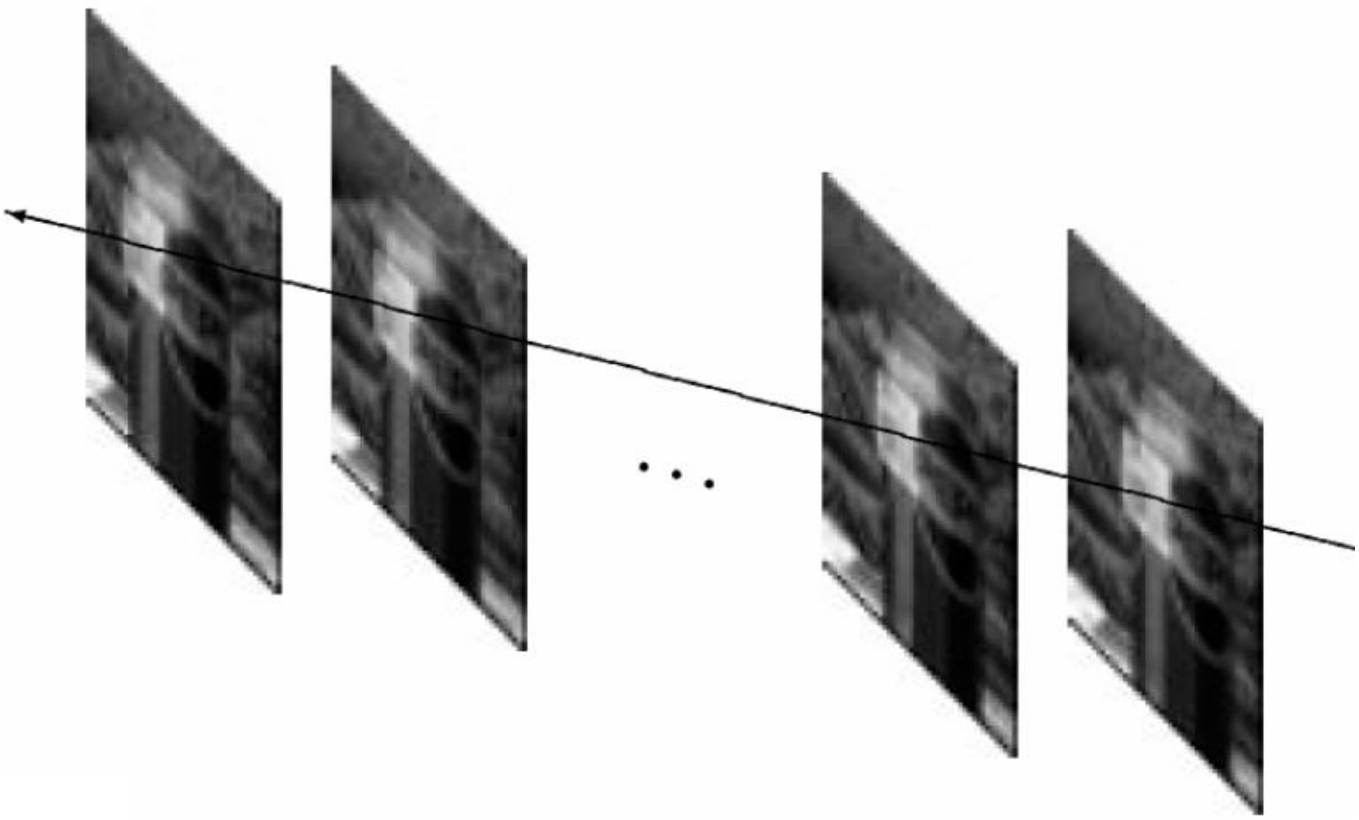
Spatial Coherence

- Neighboring points in the scene typically belong to the same surface and hence typically have similar motions.
- Since they also project to nearby pixels in the image, we expect spatial coherence in image flow.



Temporal Persistence

- The image motion of a surface patch changes gradually over time



Simple Tracking Approach

- Minimize brightness difference

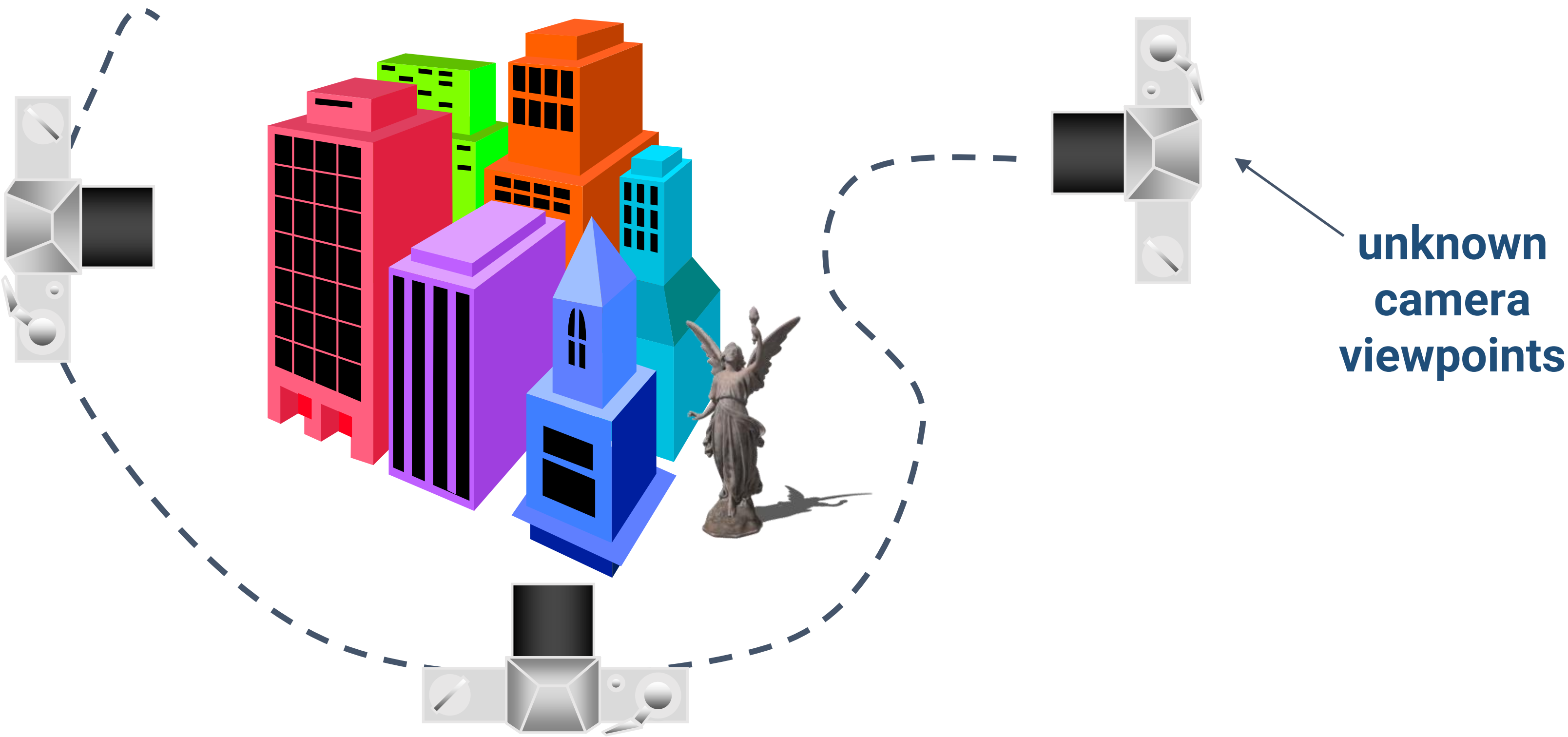
$$E(u,v) = \sum_{x,y} \left(I(x+u, y+v) - T(x,y) \right)^2$$

- For each offset (u, v) compute E(u, v)
- Choose (u, v) which minimizes E(u, v)
- Problems:
  - Not efficient
  - Only sub-pixel accuracy

There are more efficient algorithms (e.g. Lucas-Kanade) for tracking

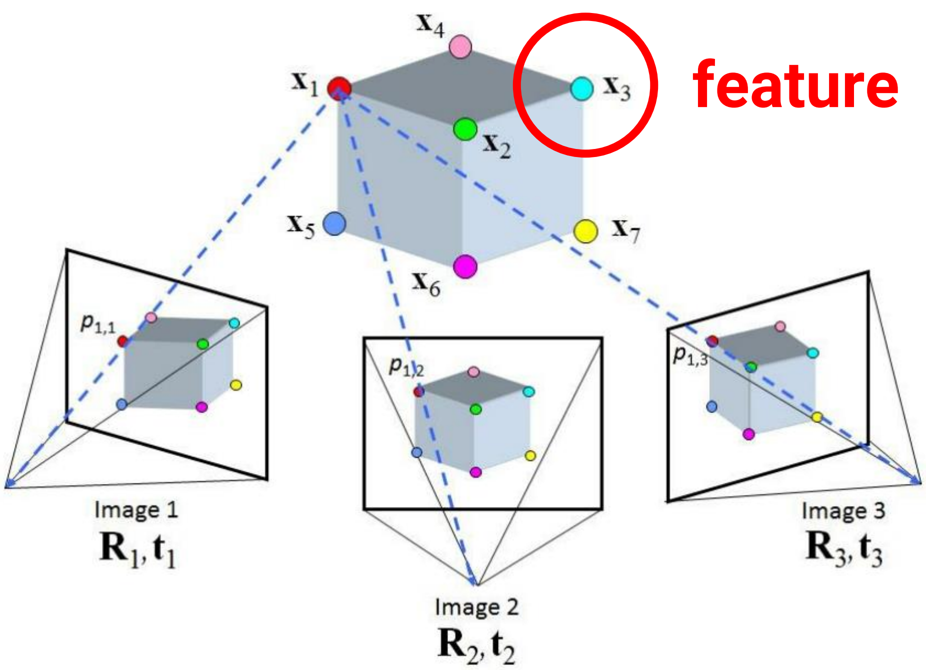
Back to the Matchmove Problem

We need to reconstruct the **camera path**
We also need to reconstruct the **(partial) scene geometry**



Back to the Matchmove Problem (cont.)

- For the scene geometry, we only recover the 3D position of **feature points**

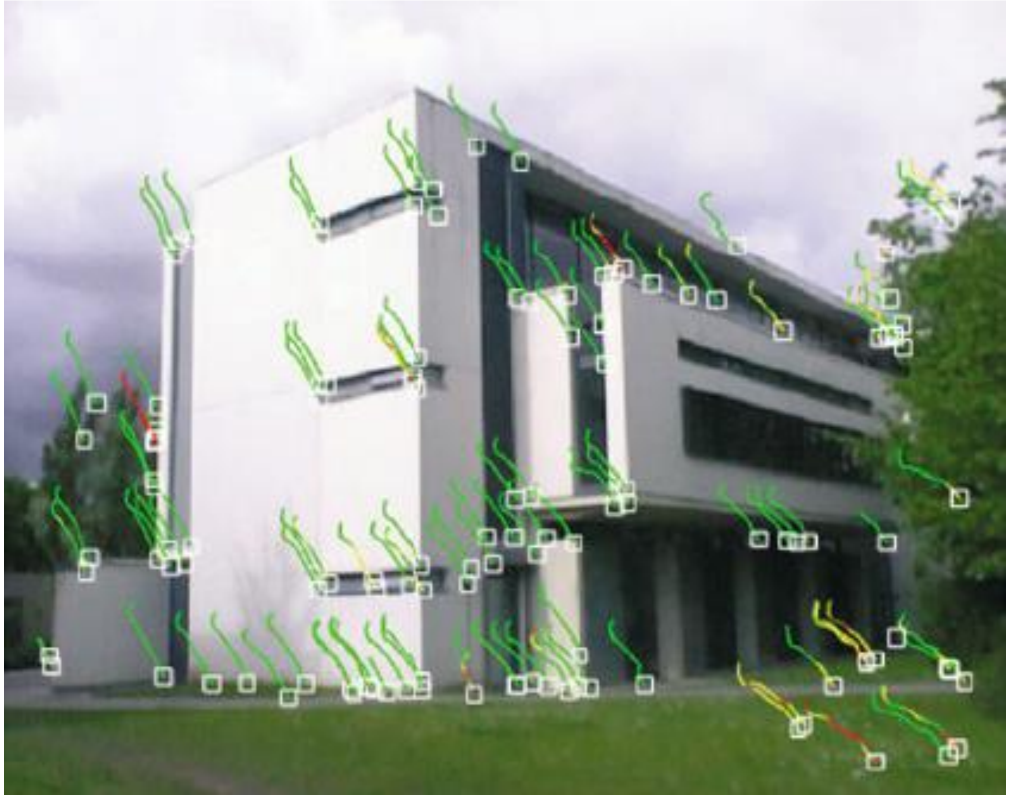


Matchmove Pipeline

2D feature tracking → **3D estimation** → **optimization (bundle adjust)** → **geometry fitting**

2D Feature Tracking

- Detect good features (e.g. by SIFT)
- Find correspondences between frames

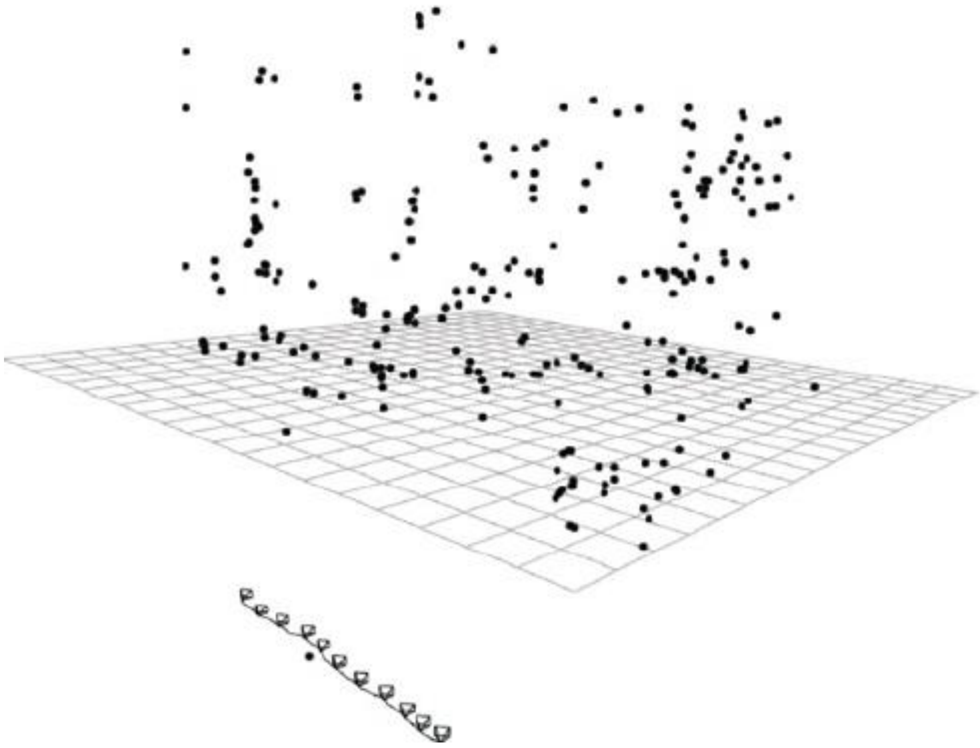


3D Estimation

- Use 2 or 3 views at a time
- Solve an optimization problem

Optimization

- Iterative refine estimates



Geometry Fitting

- Recover surface by image-based triangulation, silhouettes, or stereo

Matchmove in Blender

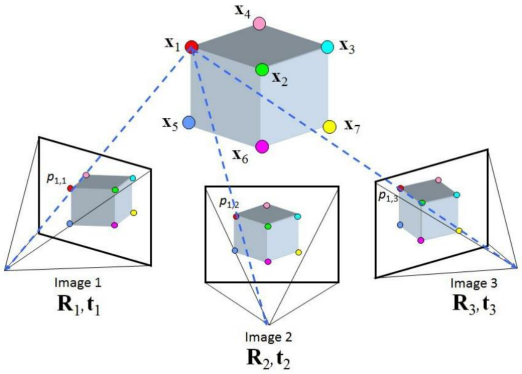
Steps

- Prepare a video
- Extract image sequence (optional)
- 3D estimation: solve and optimize camera motion and scene geometry
- Import 3D models and edit their animations
- Output video

Prepare a Video

Prepare a Video

- You can either capture your video (suggested) or download ones from the internet
- **Some useful tips**
  - It is better to have **many features** in your video
    - And the features should exist in the entire video (**especially for the ground**)
  - Not too long (if it is, subdivide it and edit each part separately)
  - Your camera should have both **translation** and **rotation**
  - Your video should have **large parallax**



Bad example

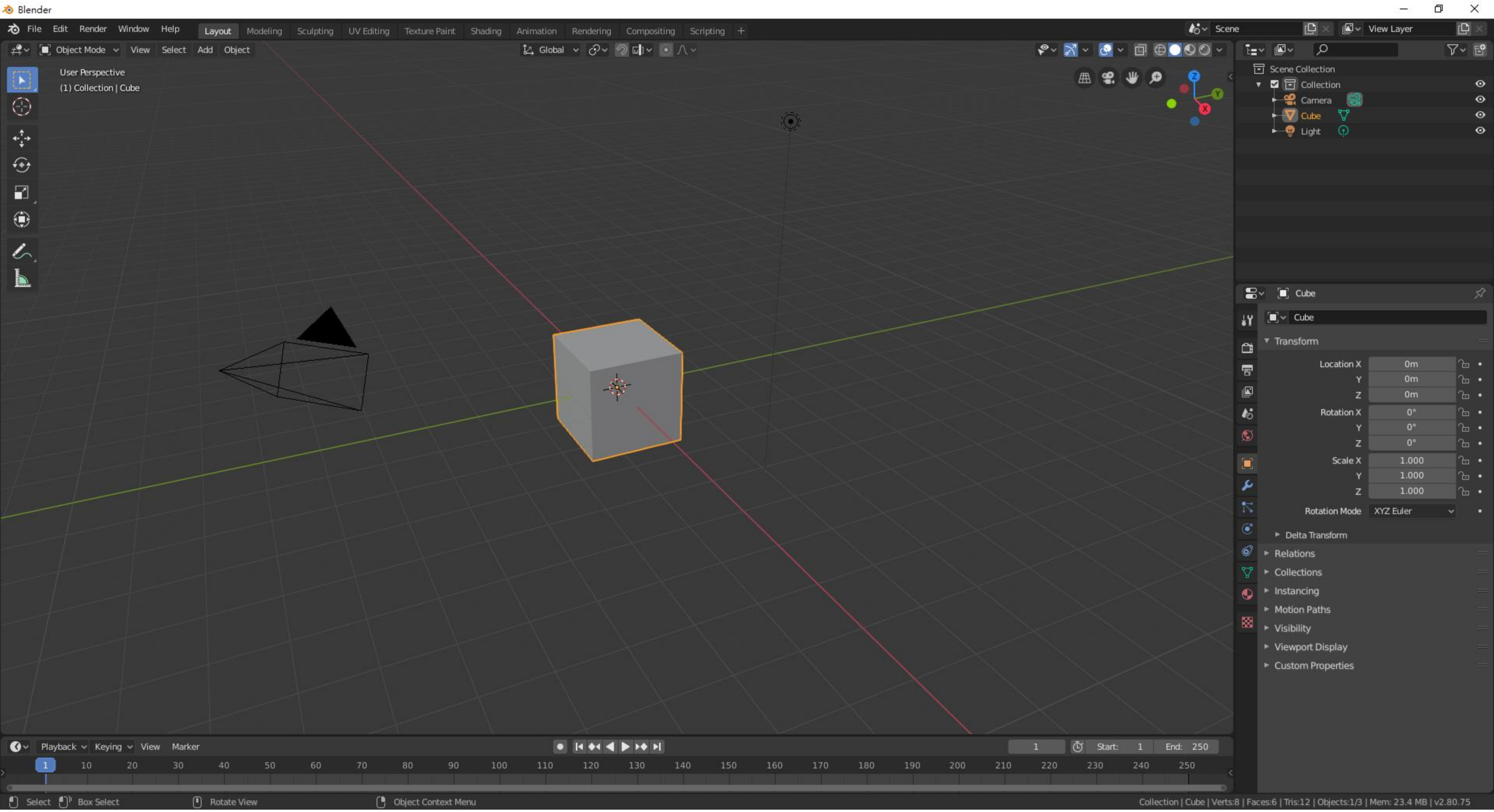
Good example

Extract Image Sequence (Optional)

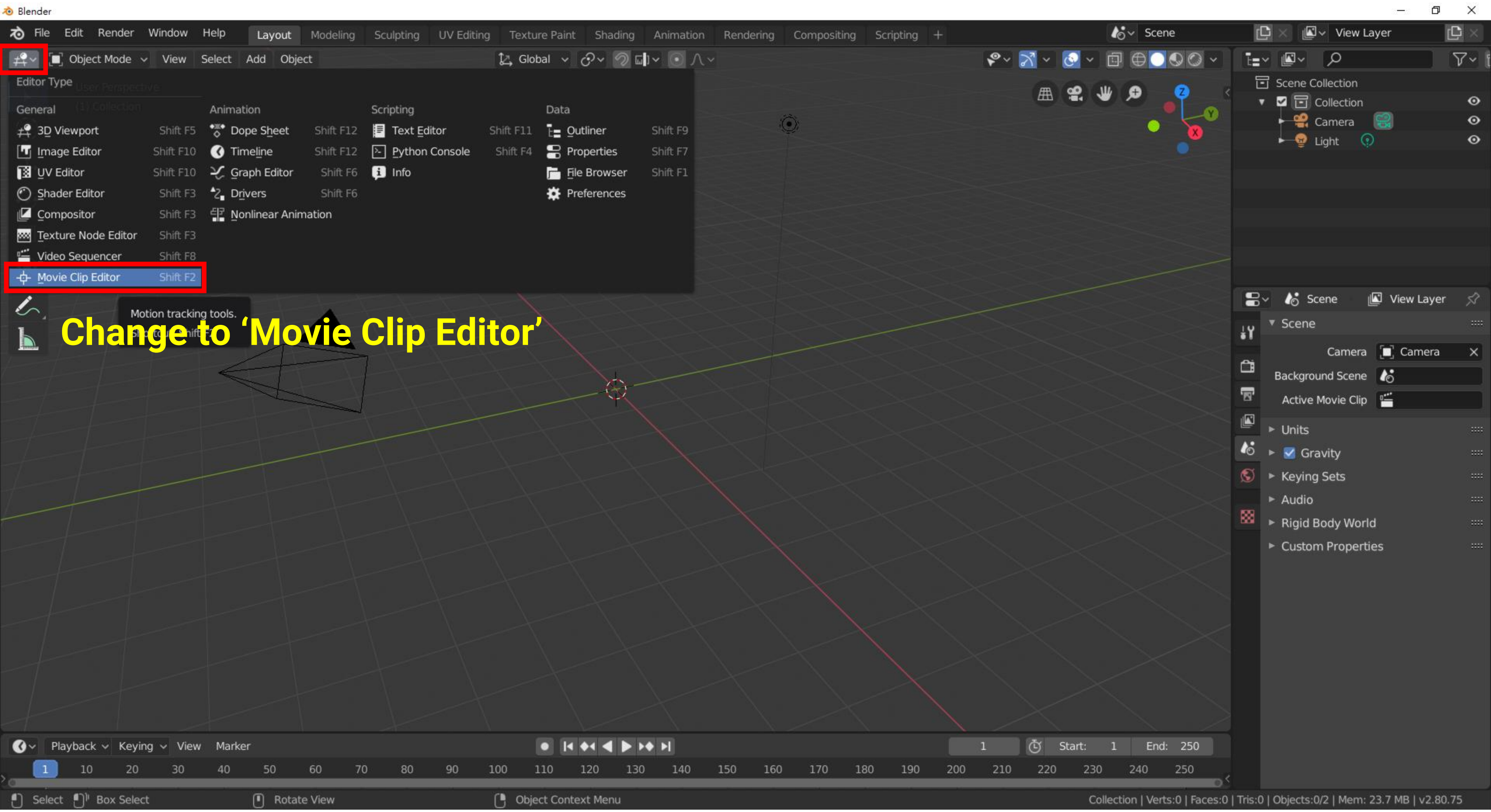
Convert a Video into Image Sequence

- **Why?**
  - Video files have compression built-in
  - Using image sequence leads to better run-time performance

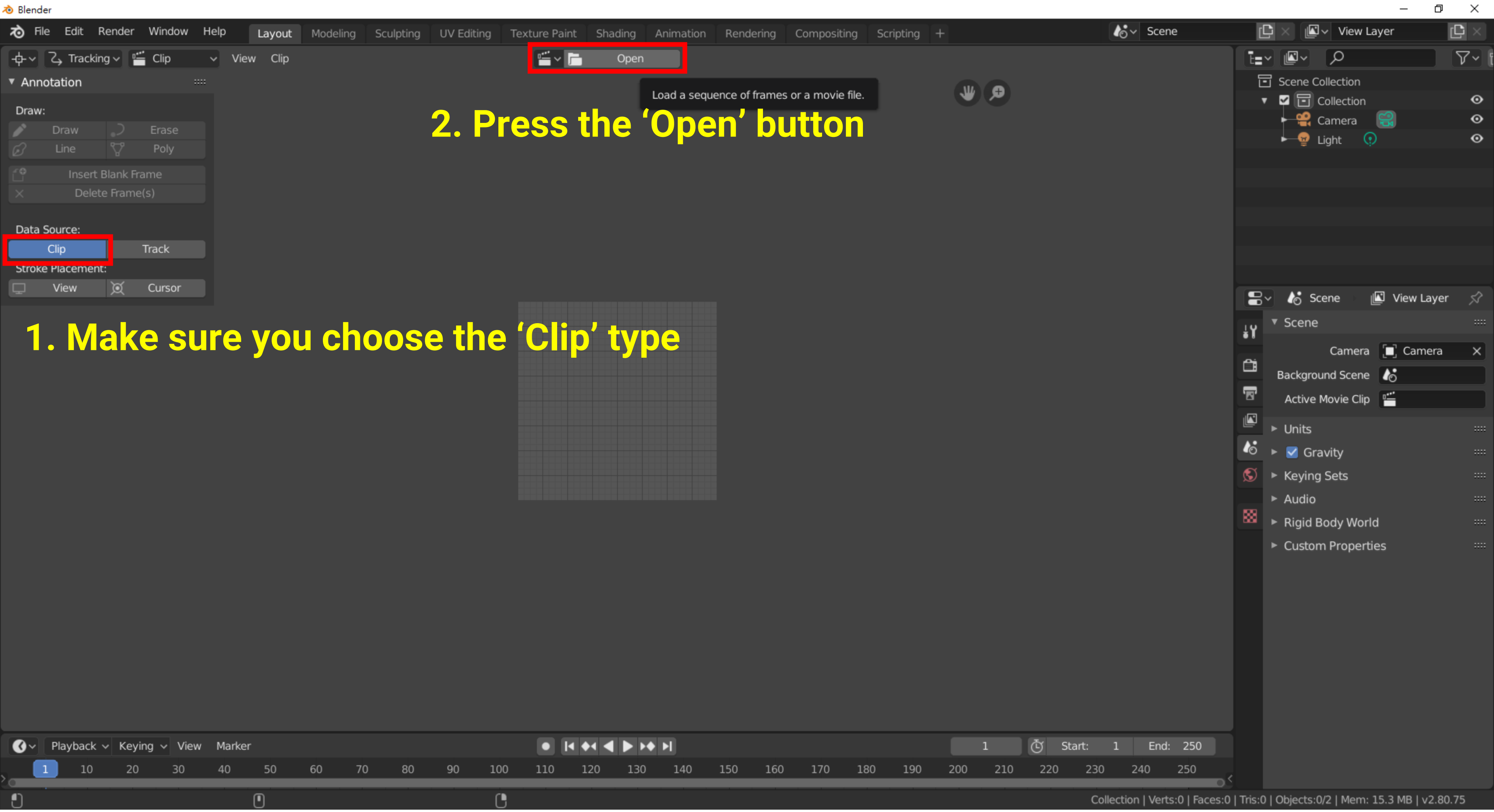
Delete unused objects



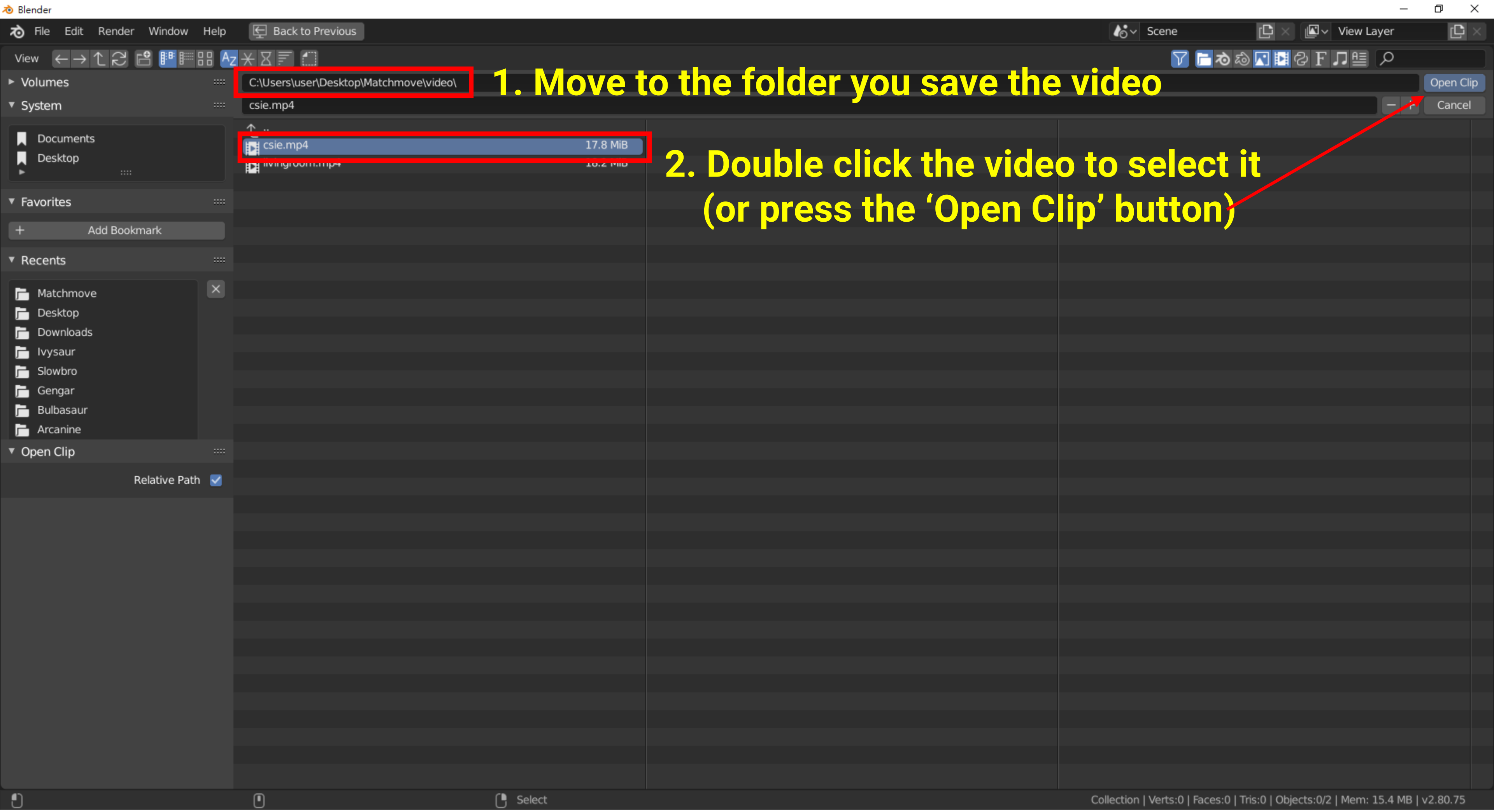
Load Your Video (1)



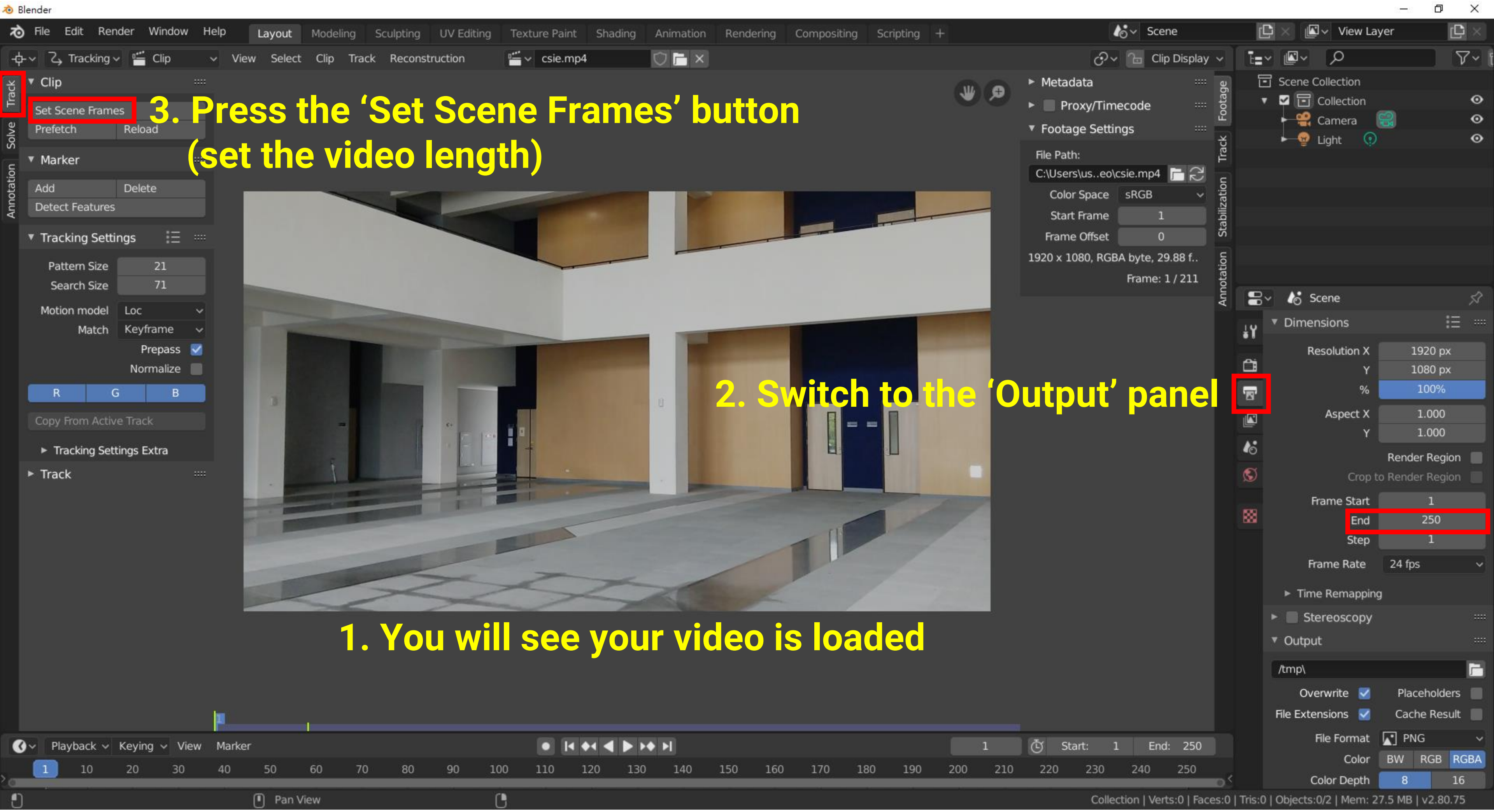
Load Your Video (2)



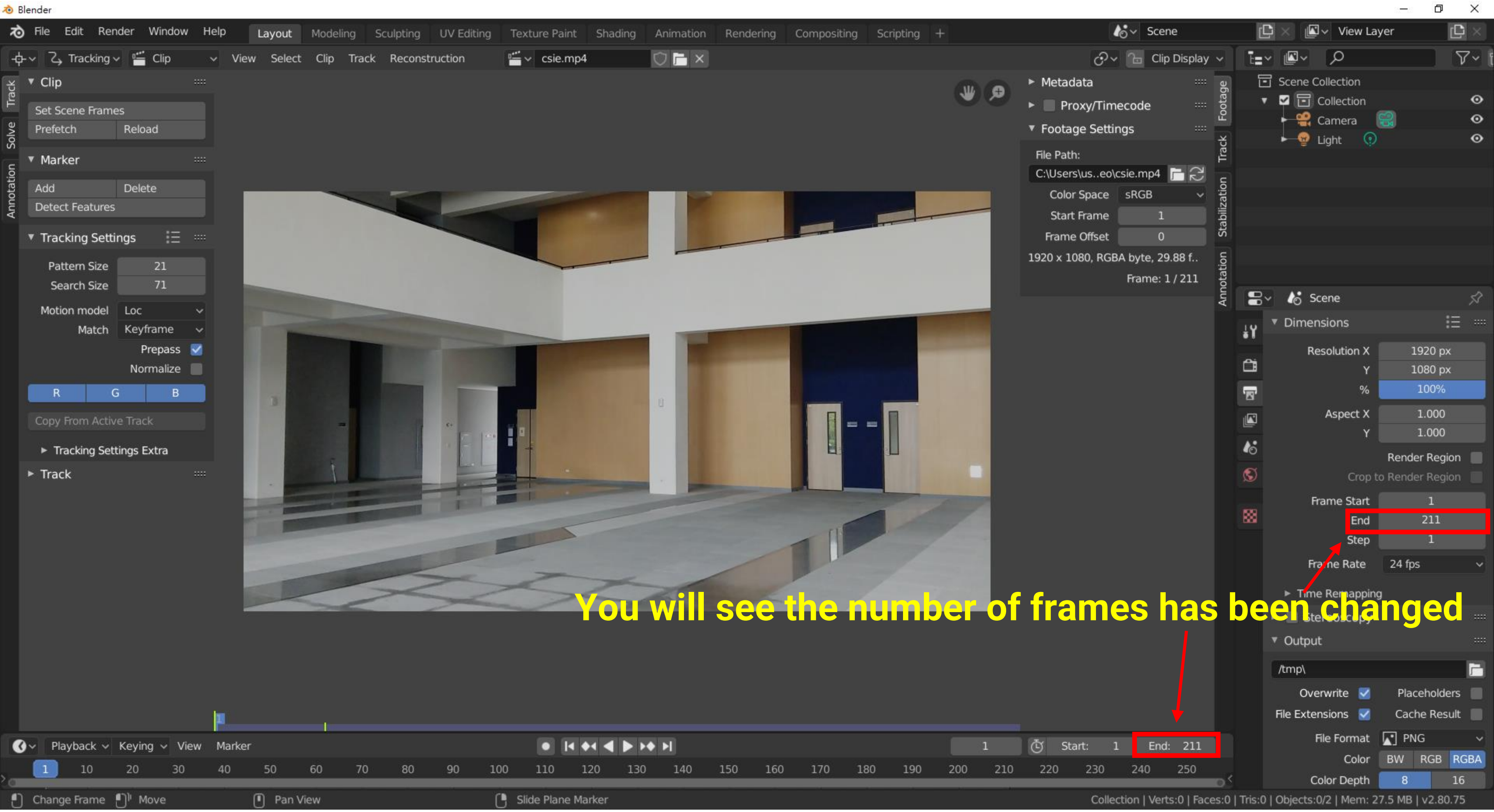
Load Your Video (3)



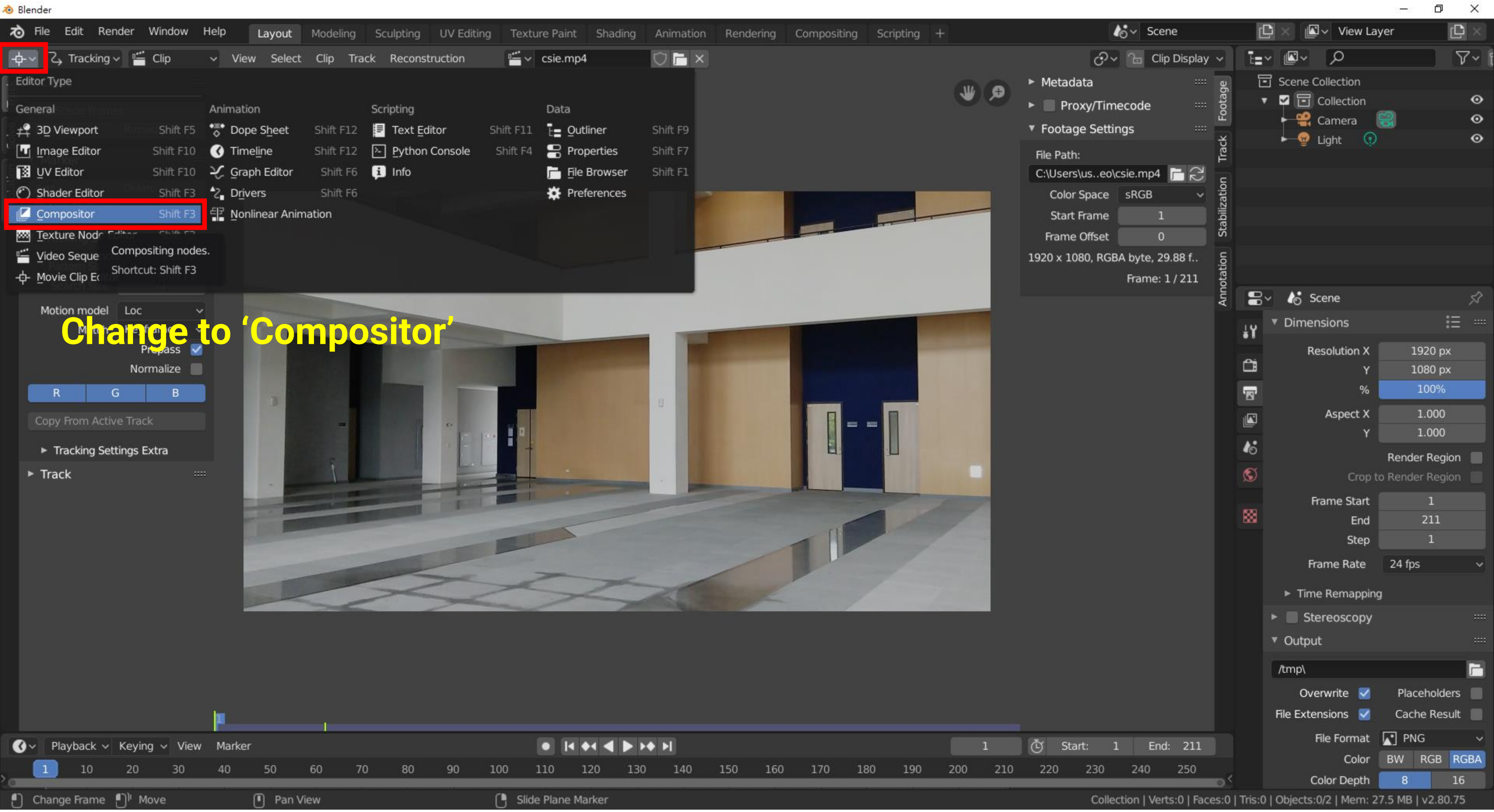
Load Your Video (4)



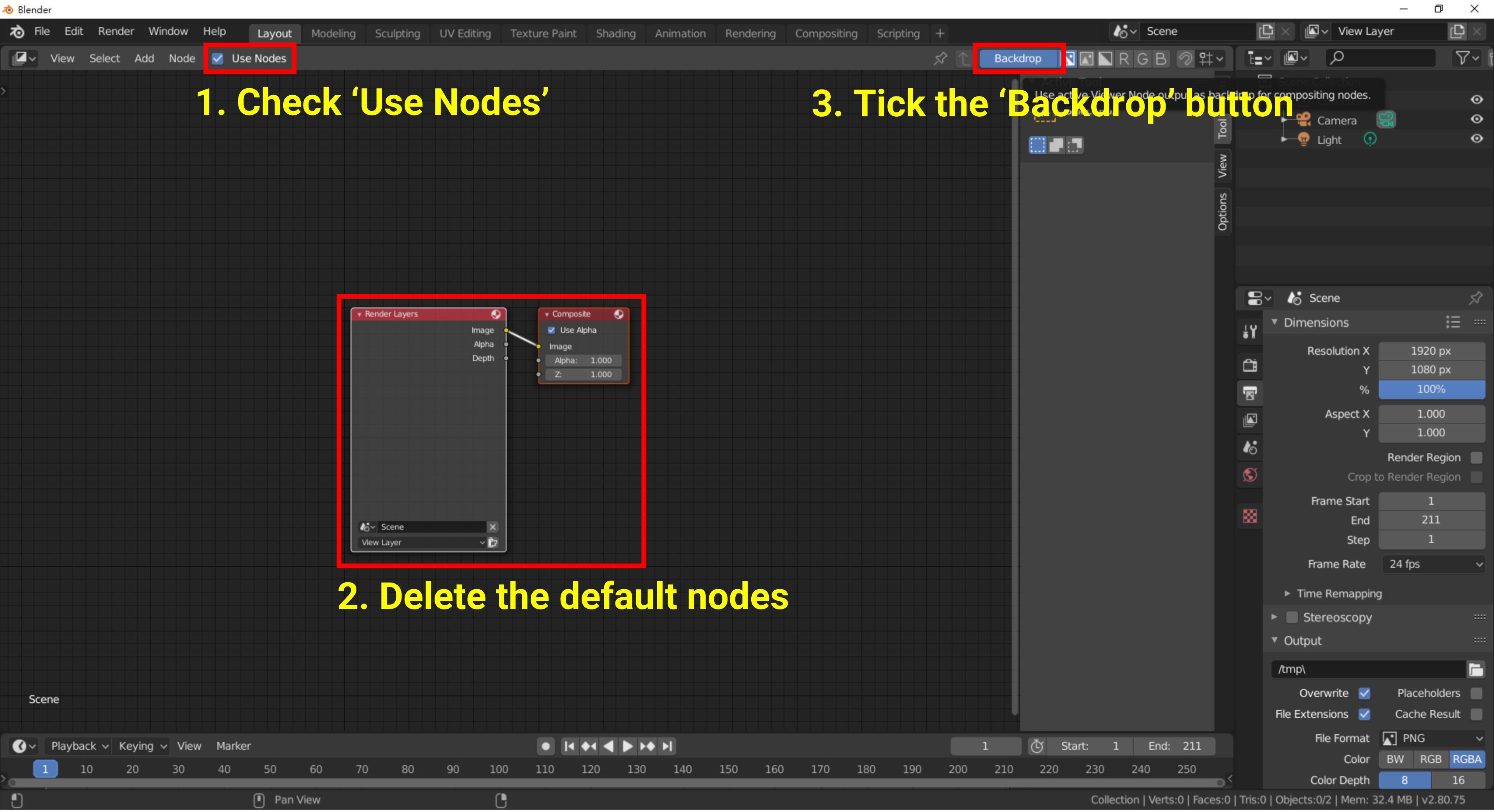
Load Your Video (5)



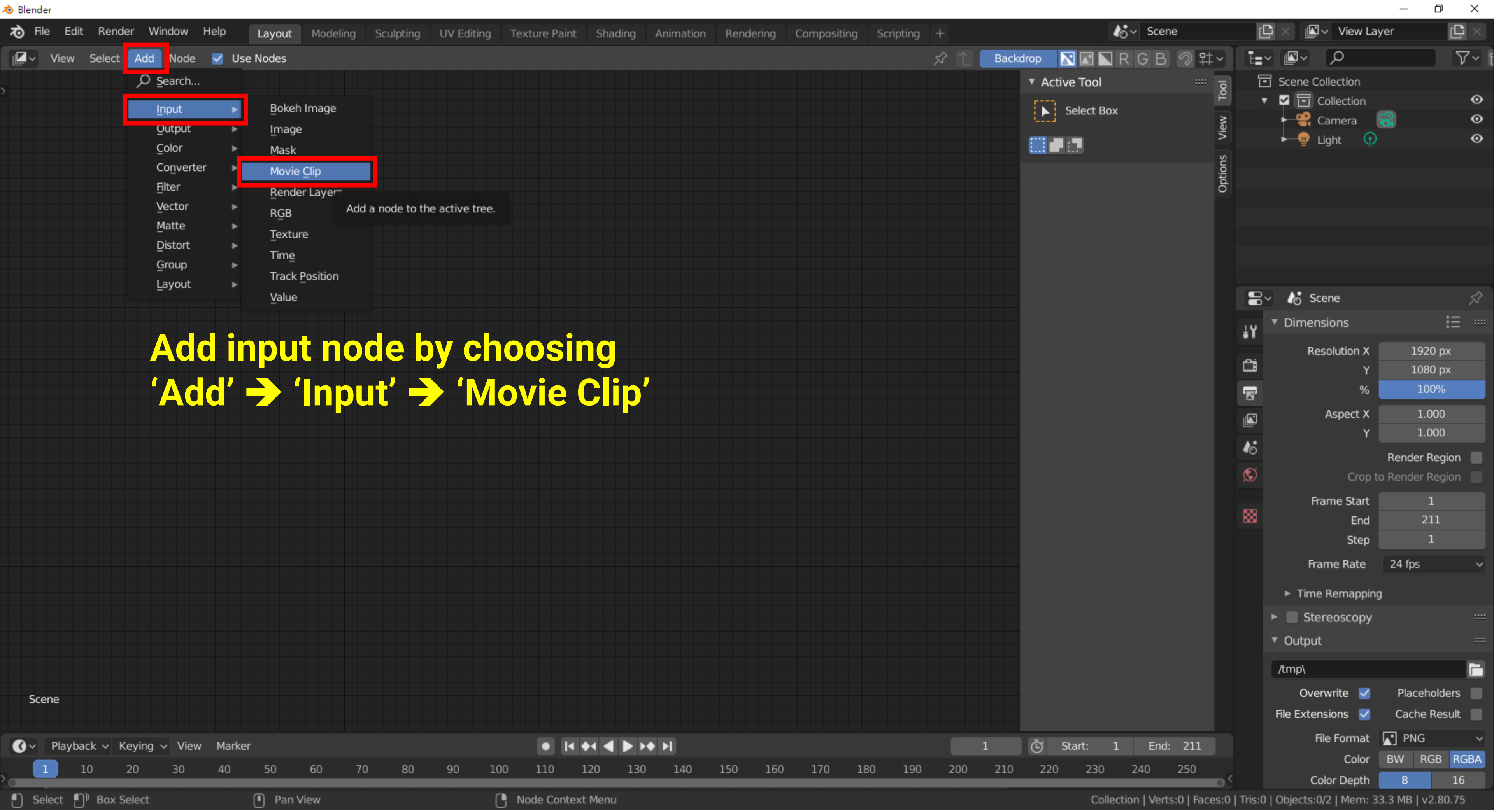
Generate Image Sequence (1)



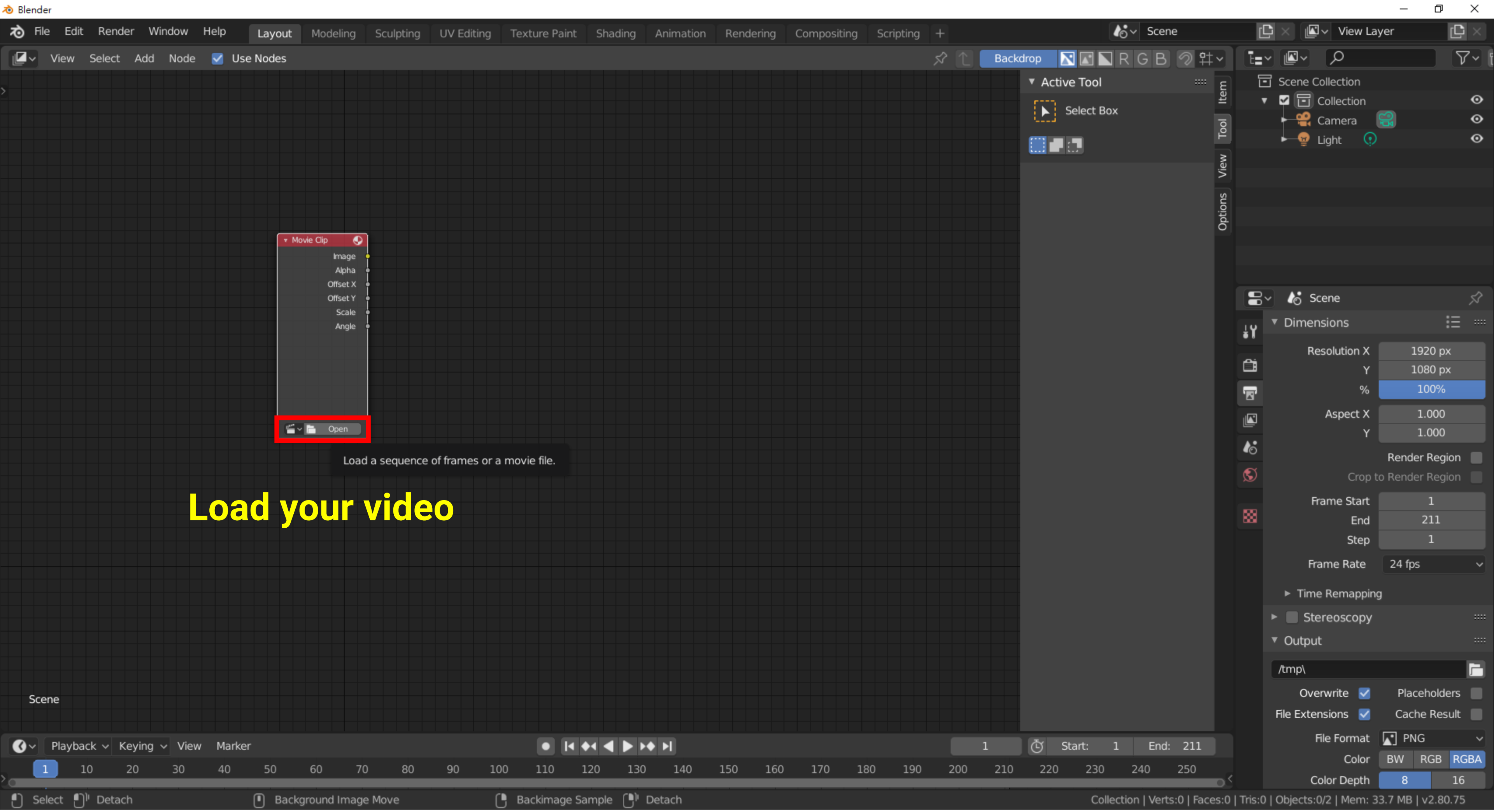
Generate Image Sequence (2)



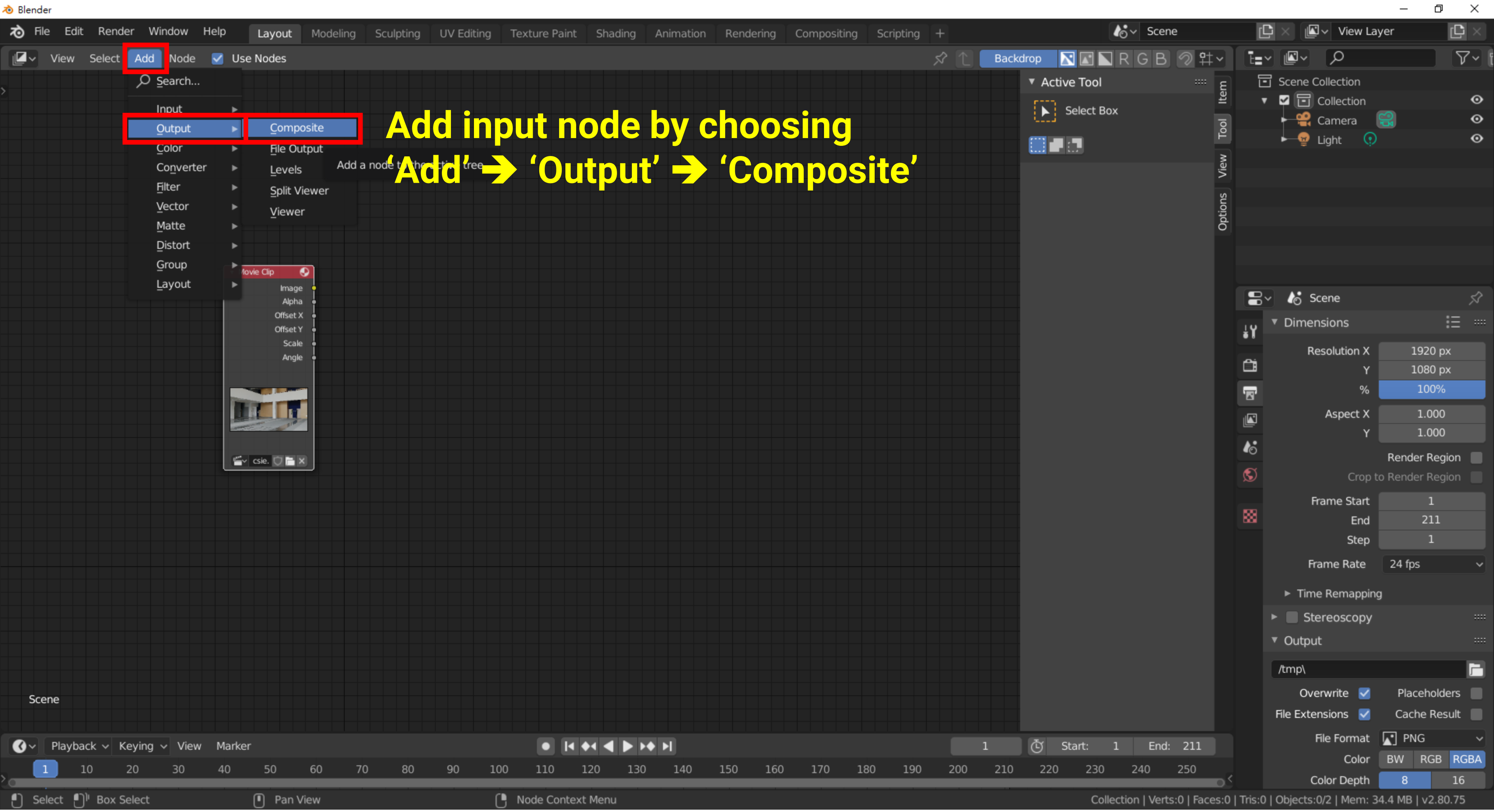
Generate Image Sequence (3)



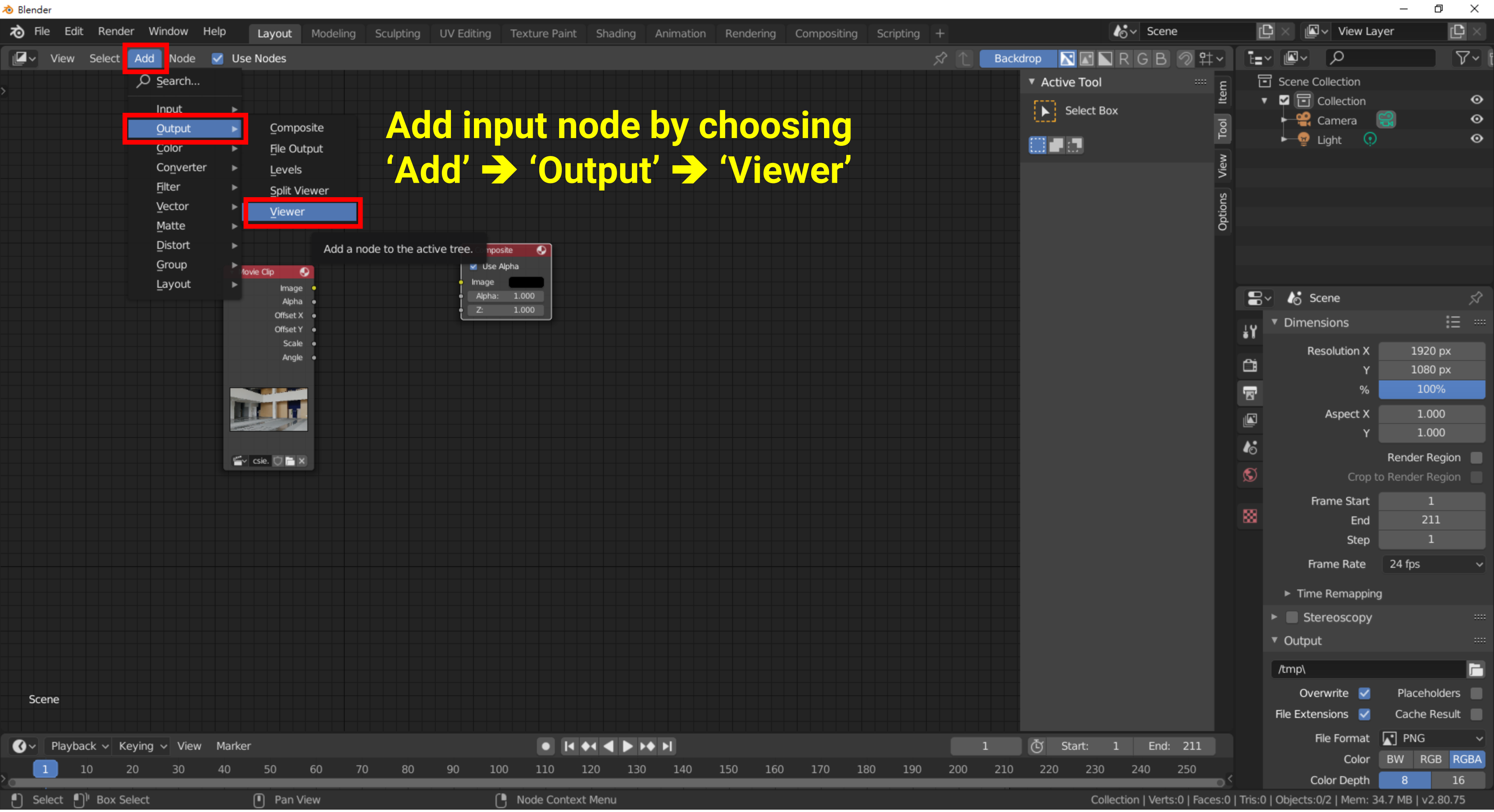
Generate Image Sequence (4)



Generate Image Sequence (5)

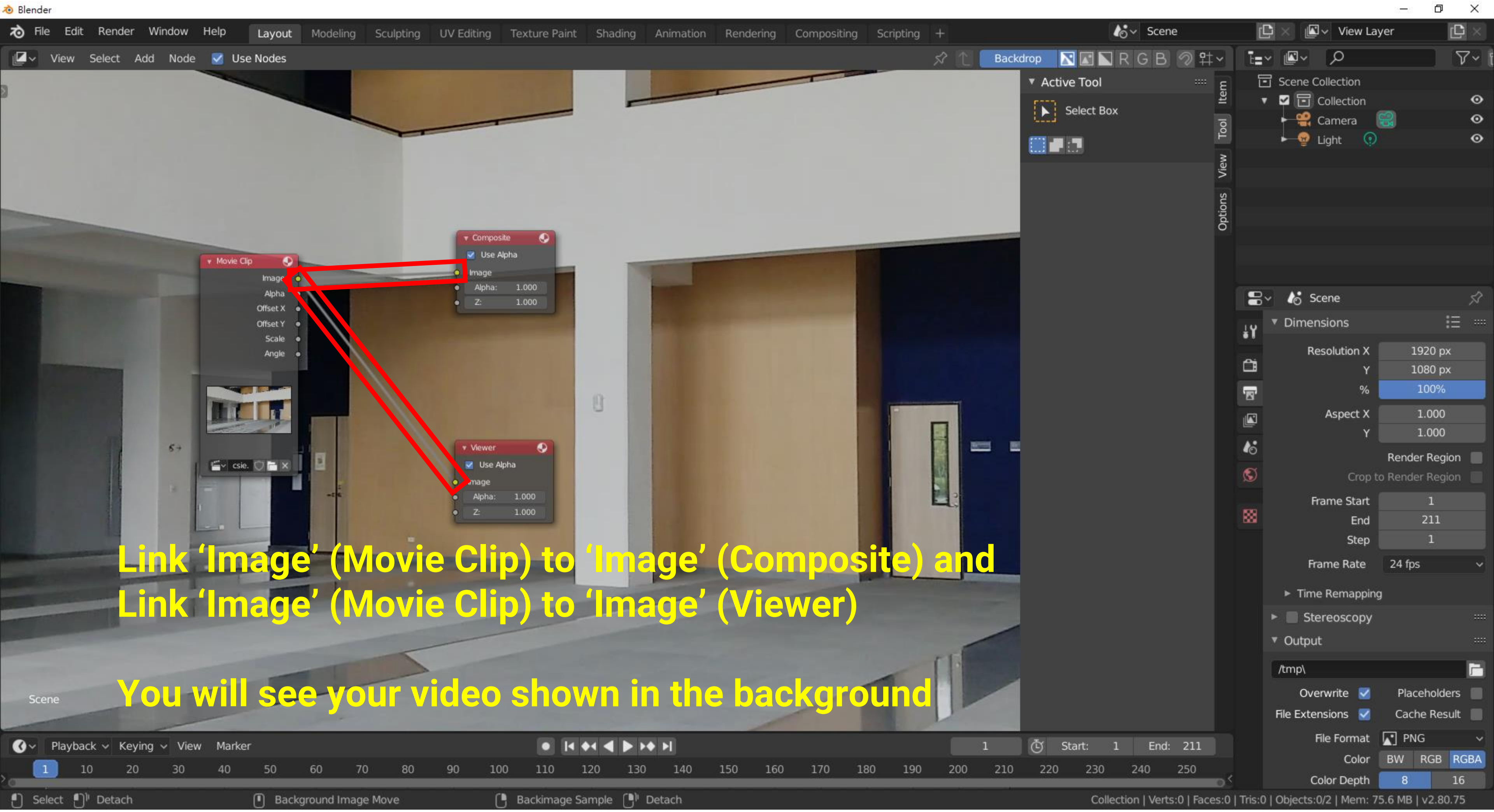


Generate Image Sequence (6)

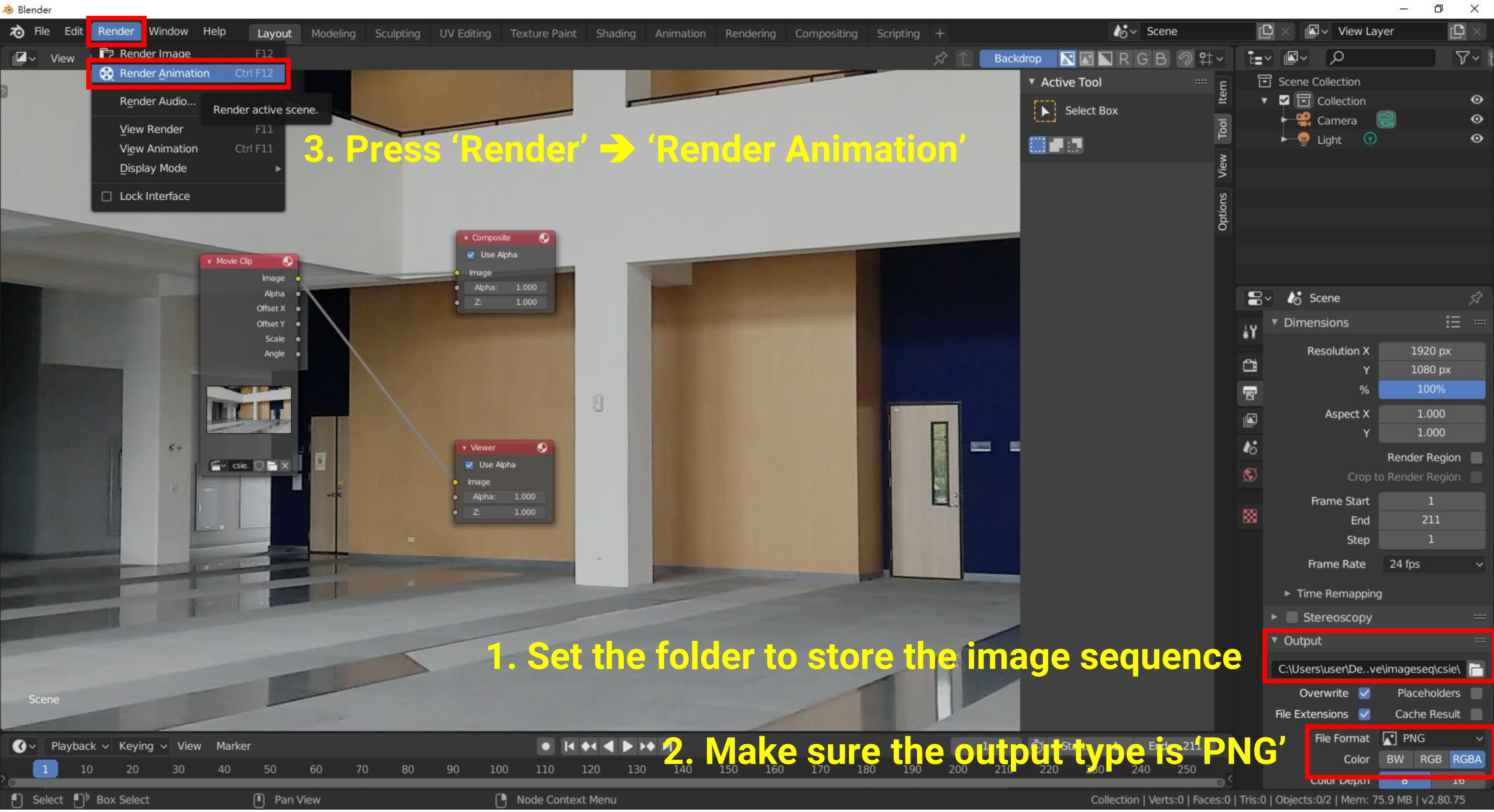


Add input node by choosing
'Add' ➔ 'Output' ➔ 'Viewer'

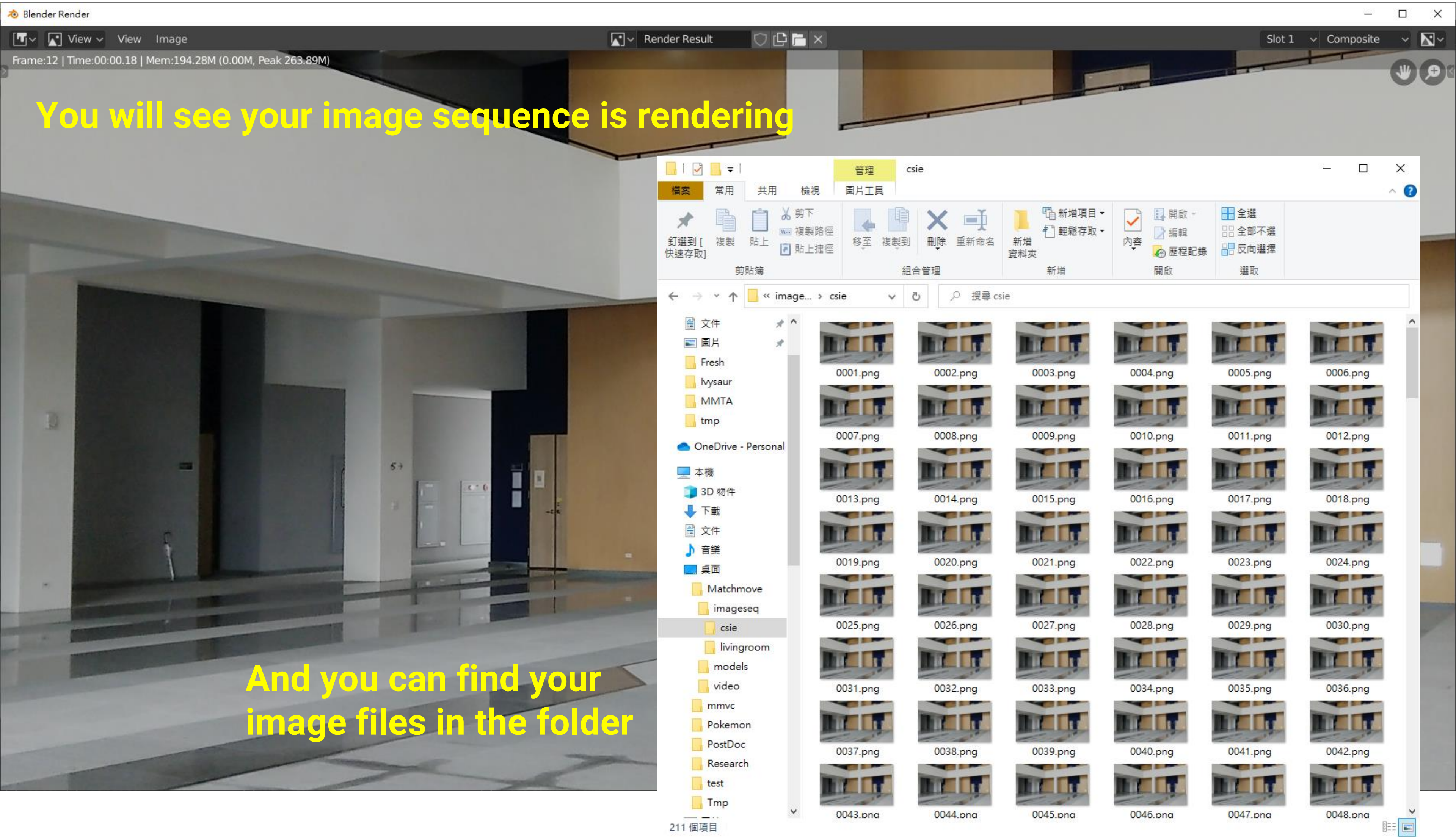
Generate Image Sequence (7)



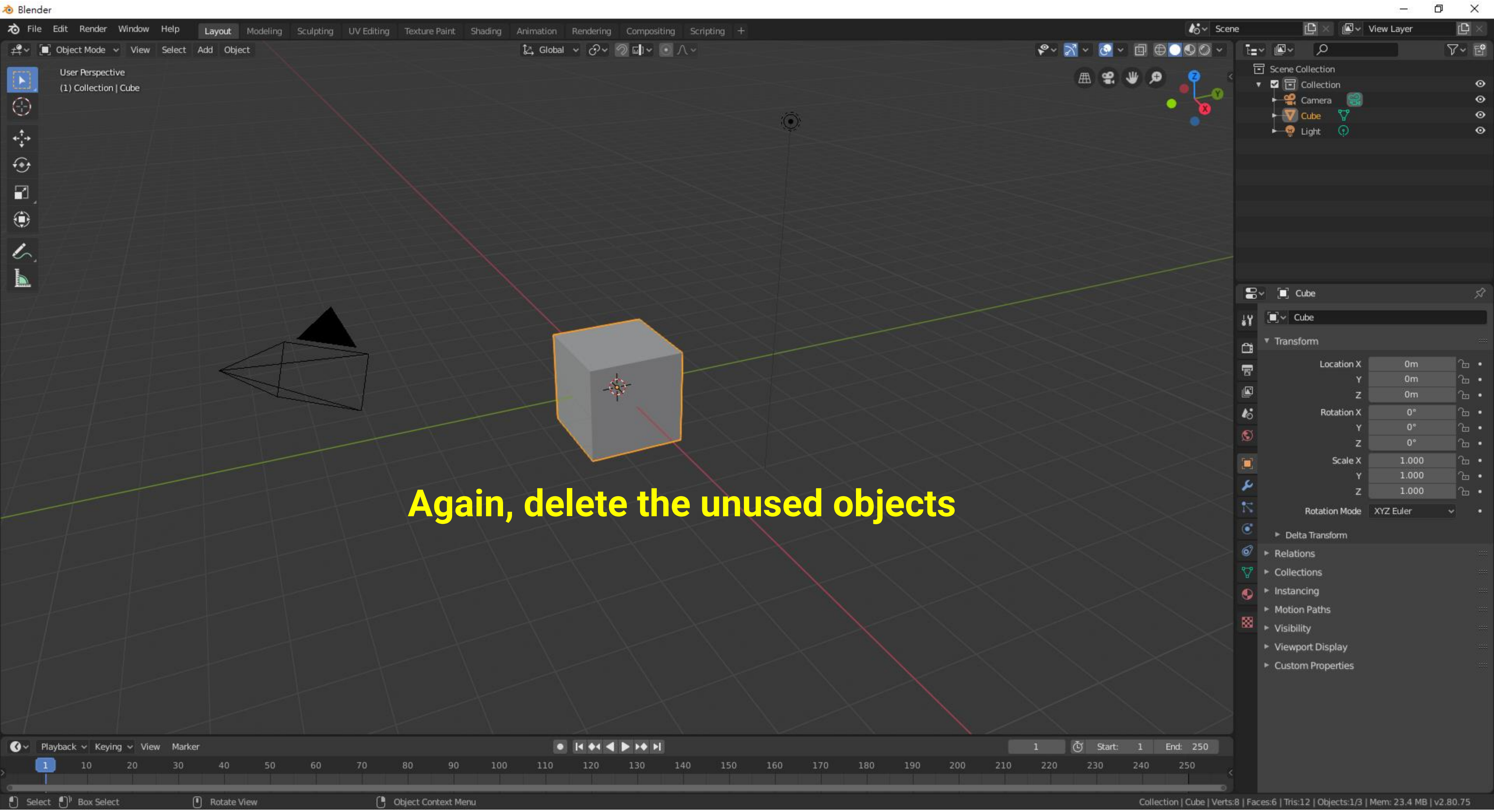
Generate Image Sequence (8)



Generate Image Sequence (9)

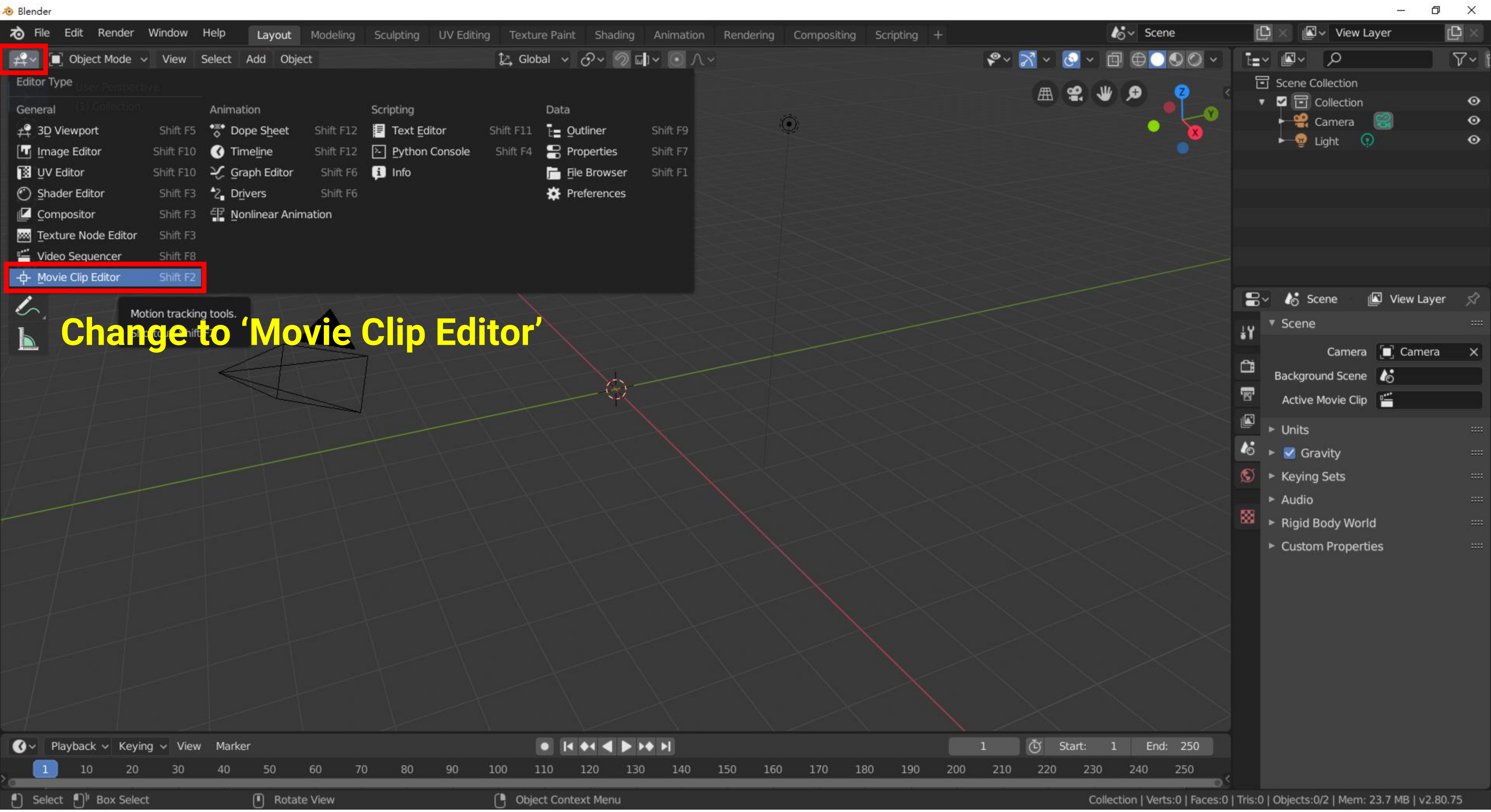


Close your Blender and Reopen It

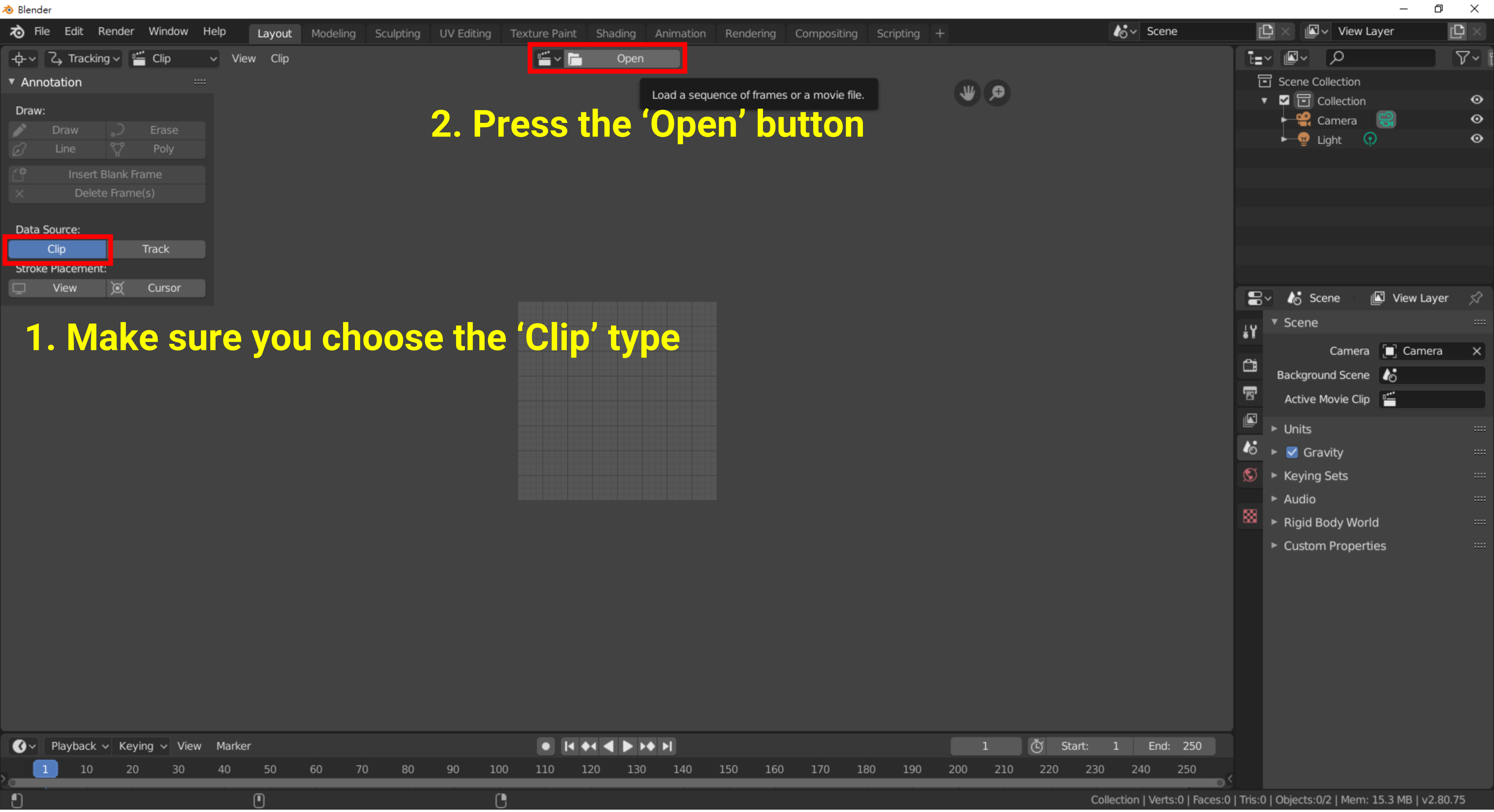


3D Estimation

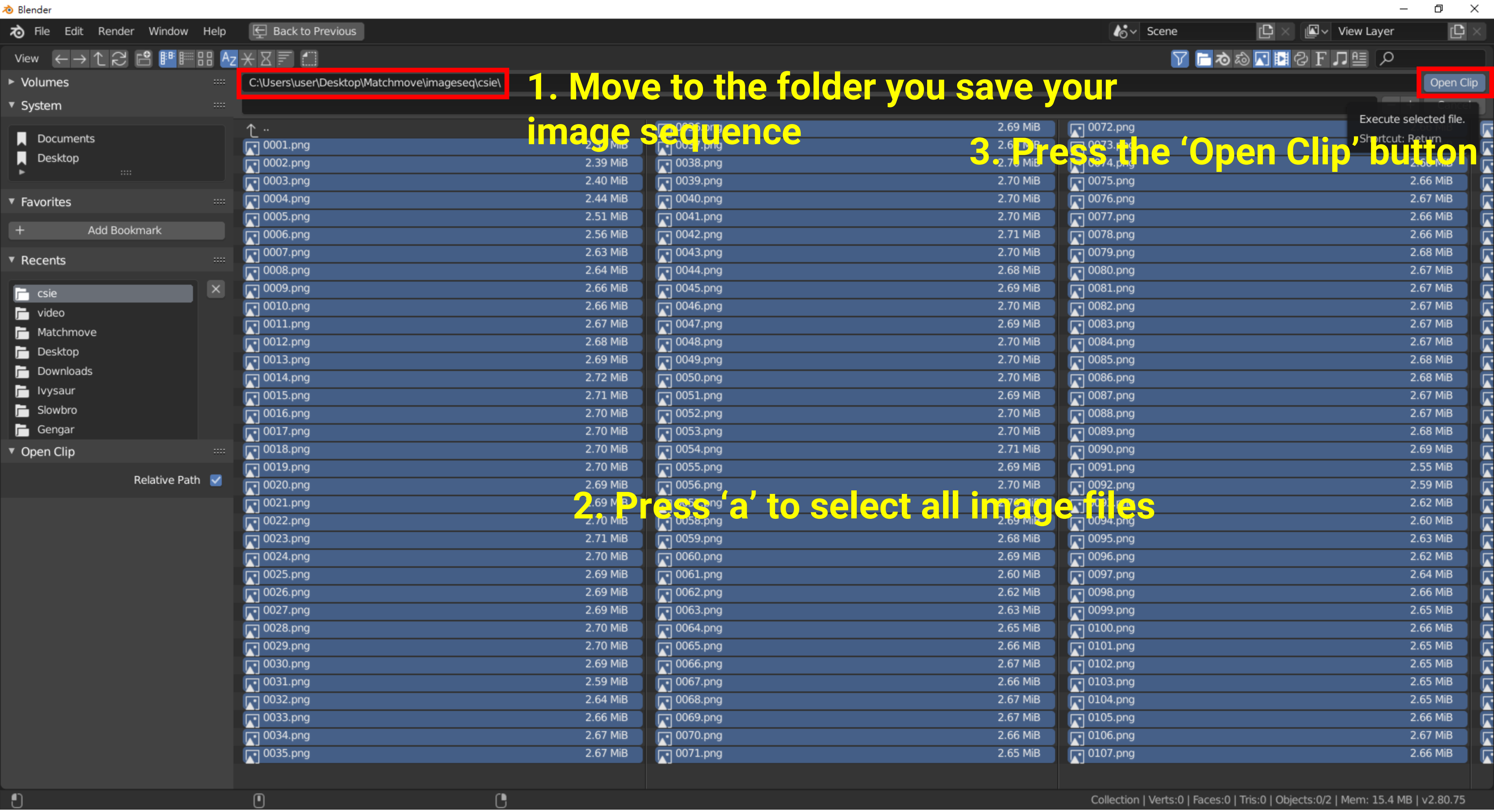
Load the Image Sequence (1)



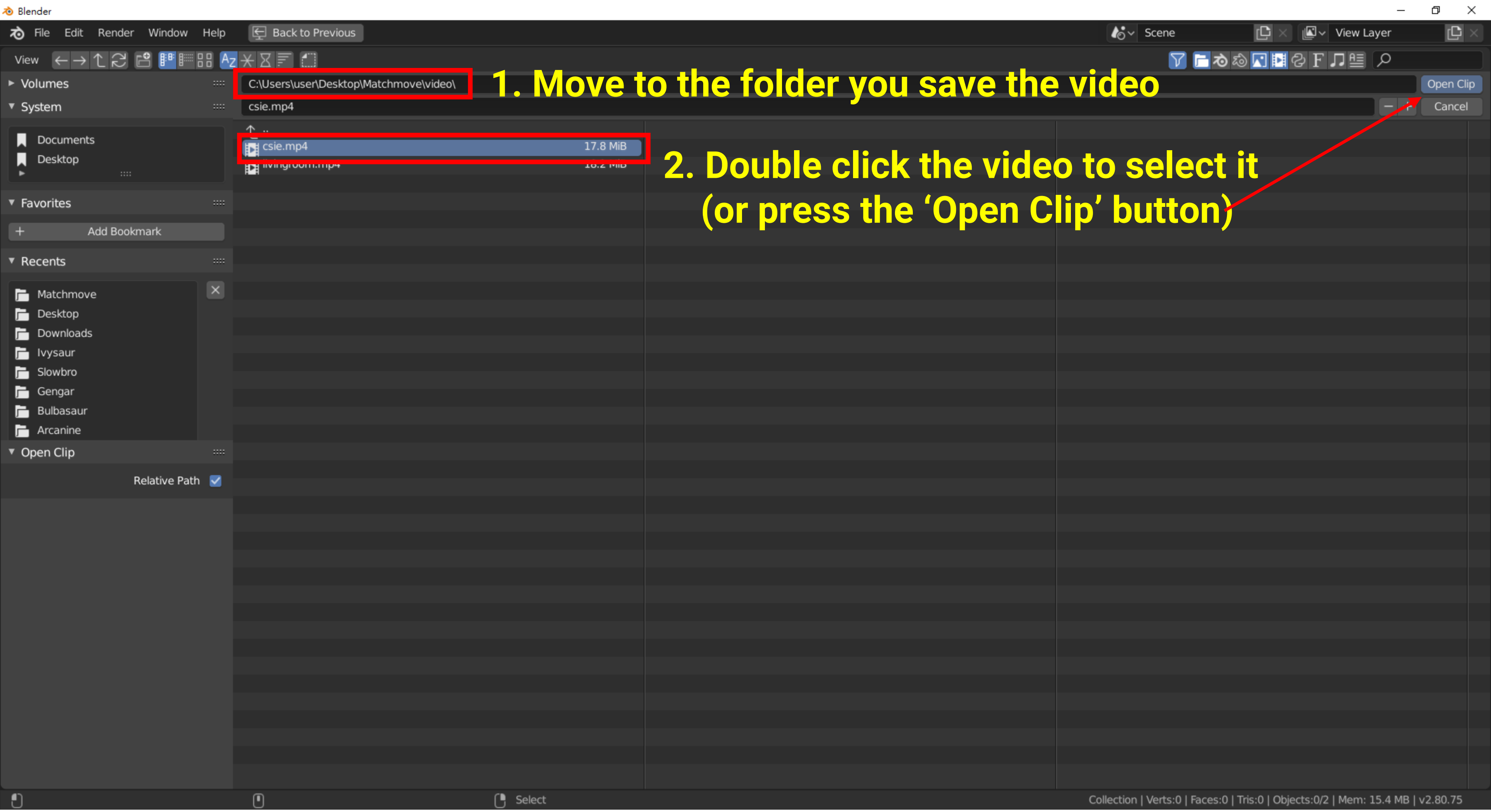
Load the Image Sequence (2)



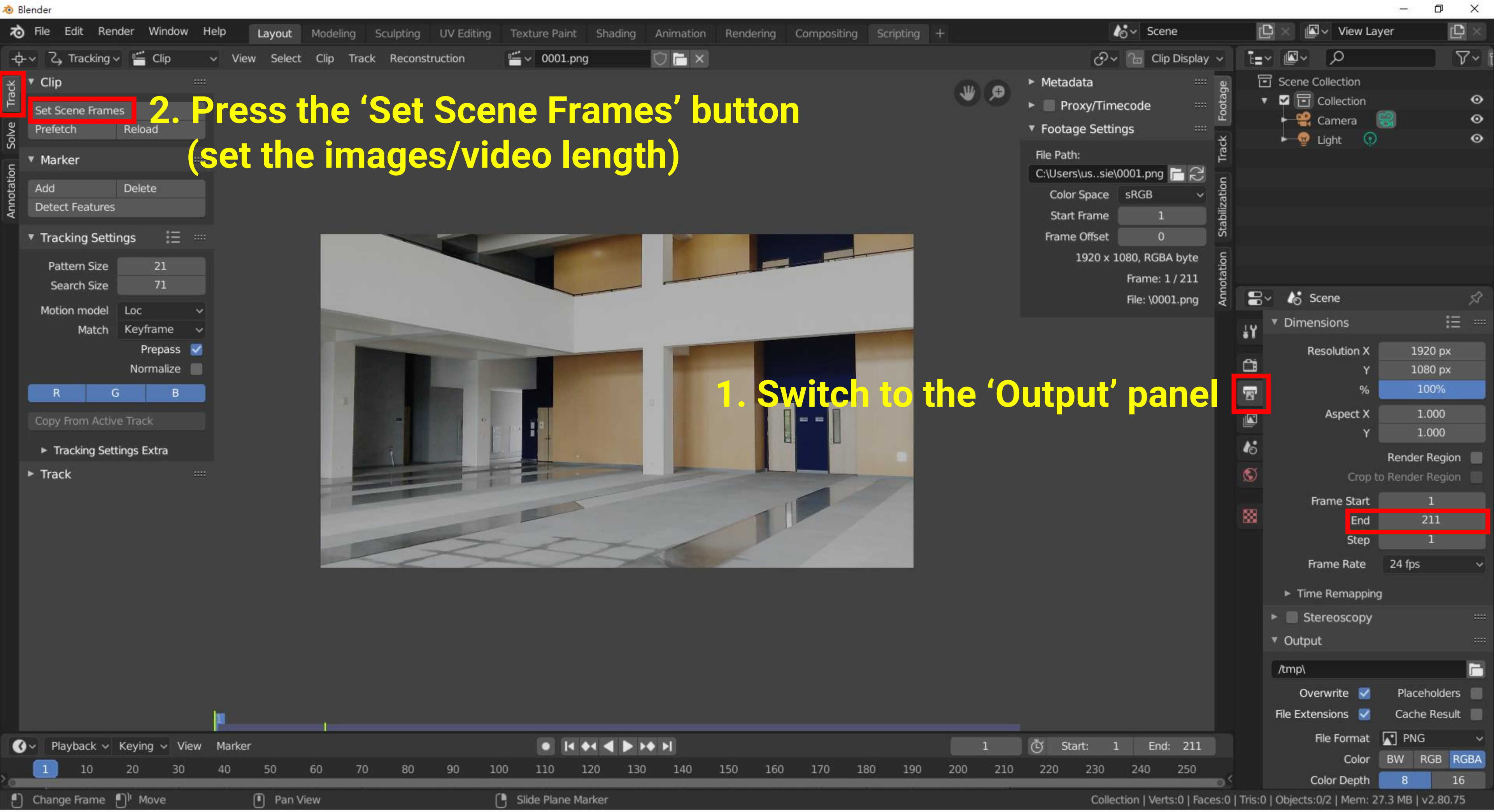
Load the Image Sequence (3)



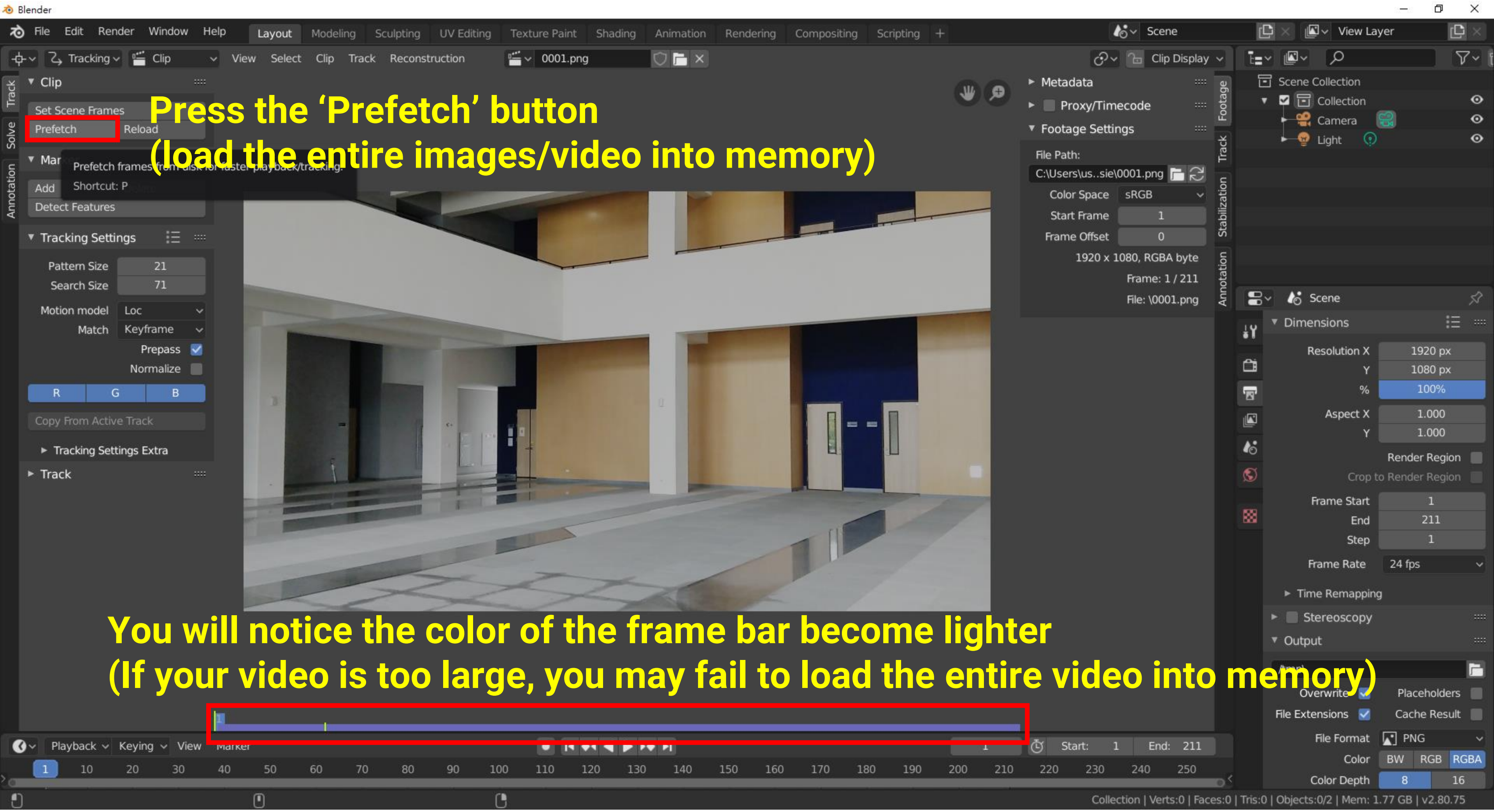
Or Load Your Video (If you skip step 1)



Set Input Data (1)



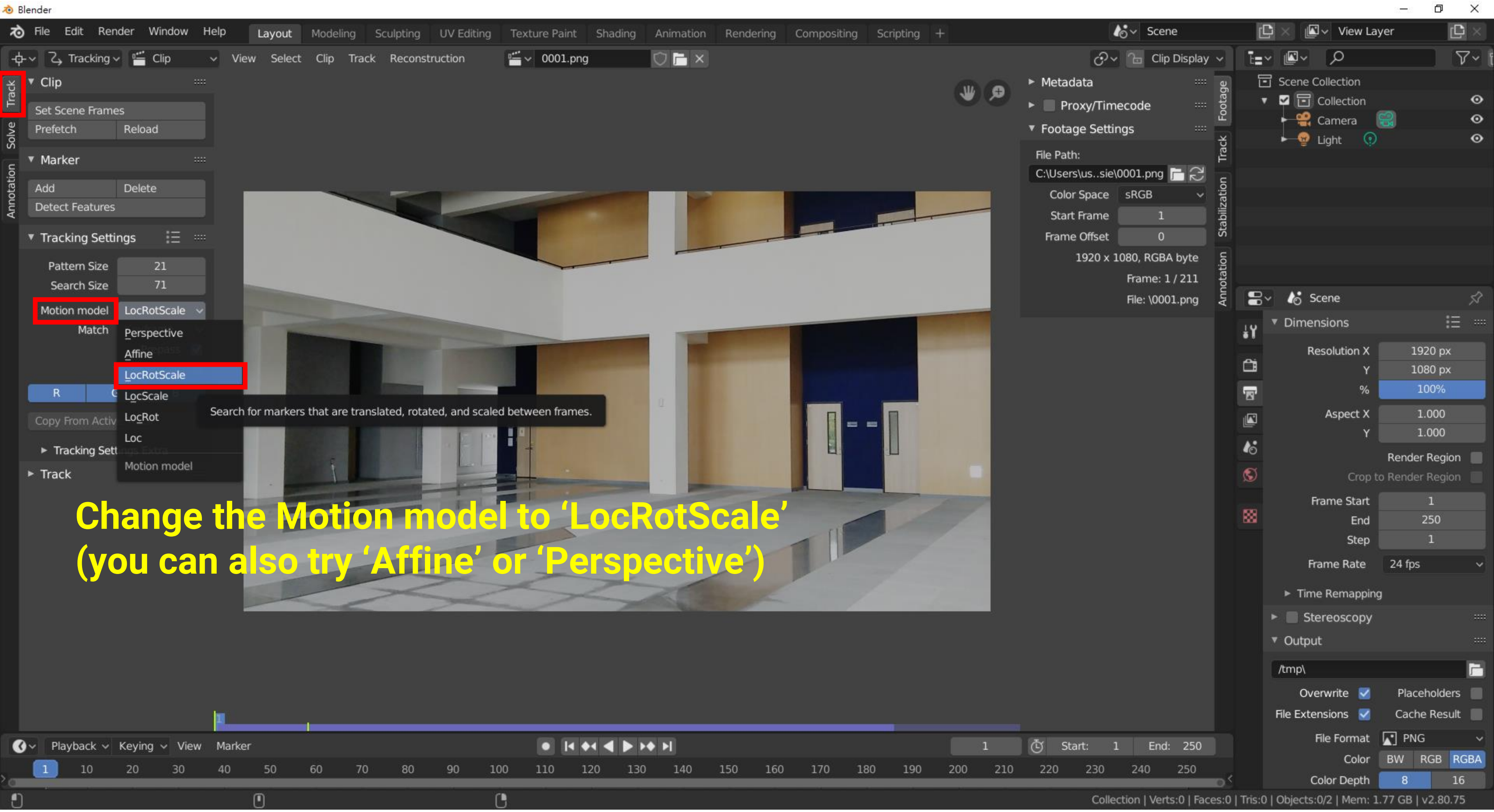
Set Input Data (2)



Press the 'Prefetch' button
(load the entire images/video into memory)

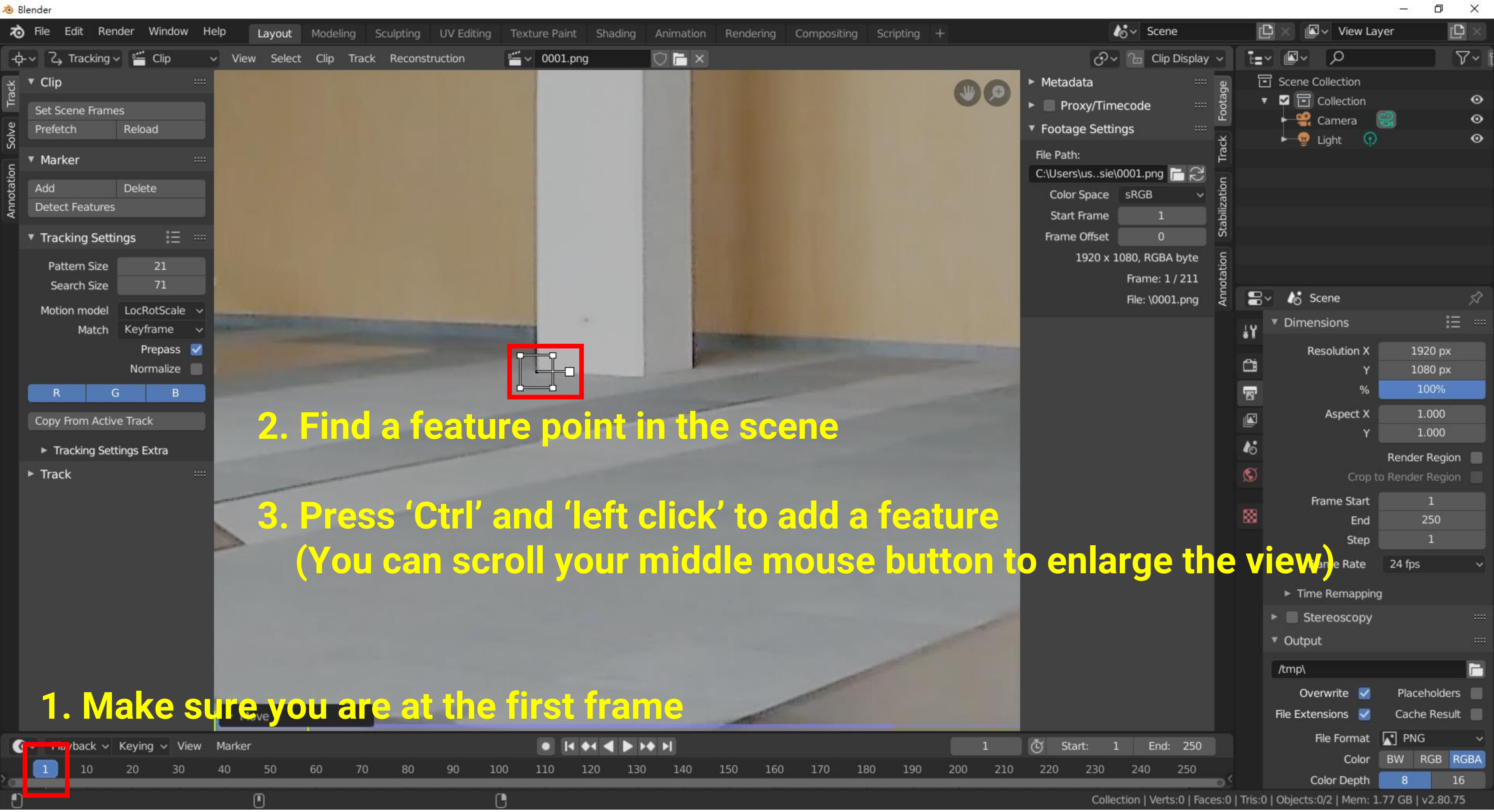
You will notice the color of the frame bar become lighter
(If your video is too large, you may fail to load the entire video into memory)

Feature Detection (1)

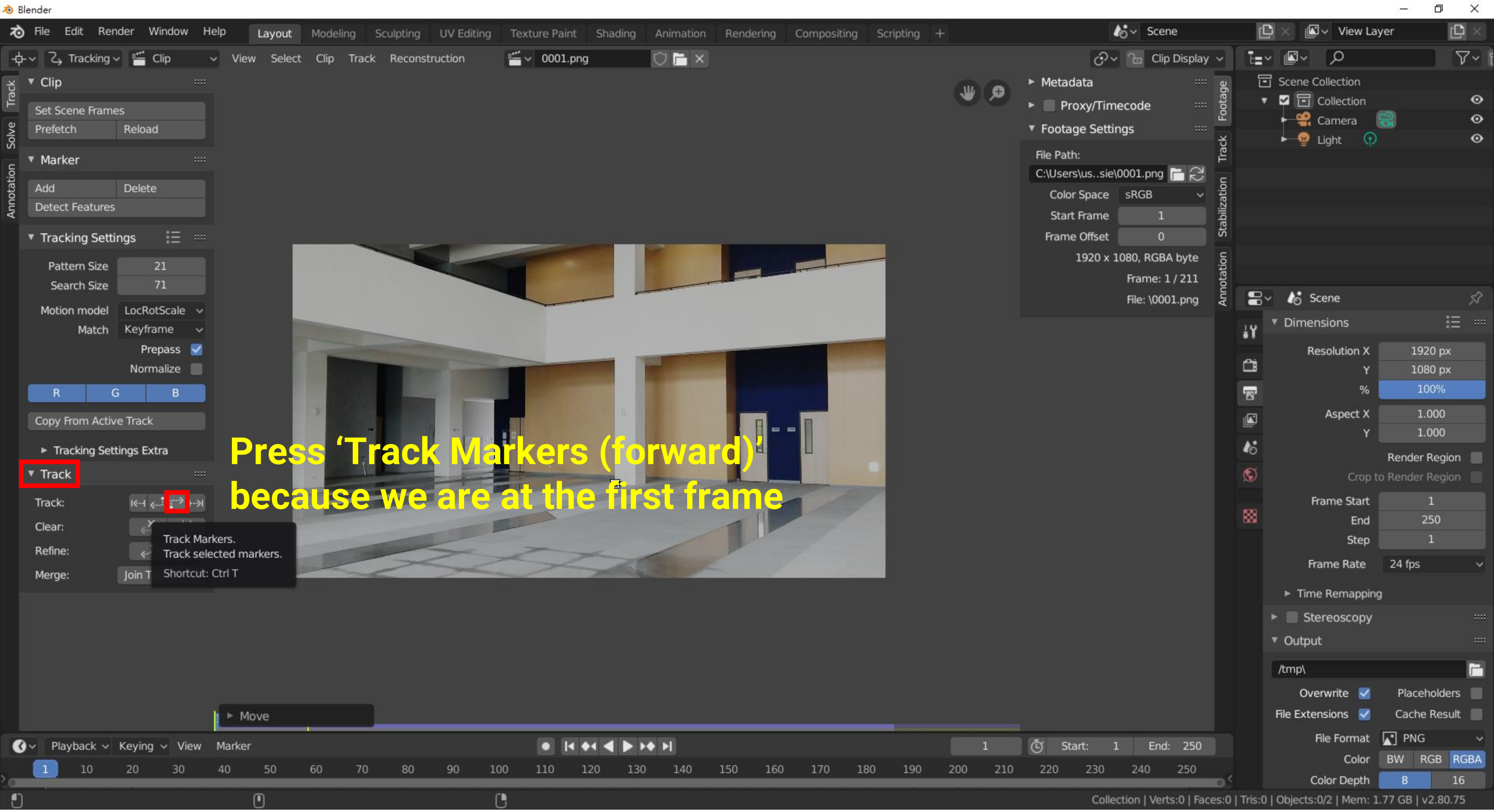


Change the Motion model to ‘LocRotScale’
(you can also try ‘Affine’ or ‘Perspective’)

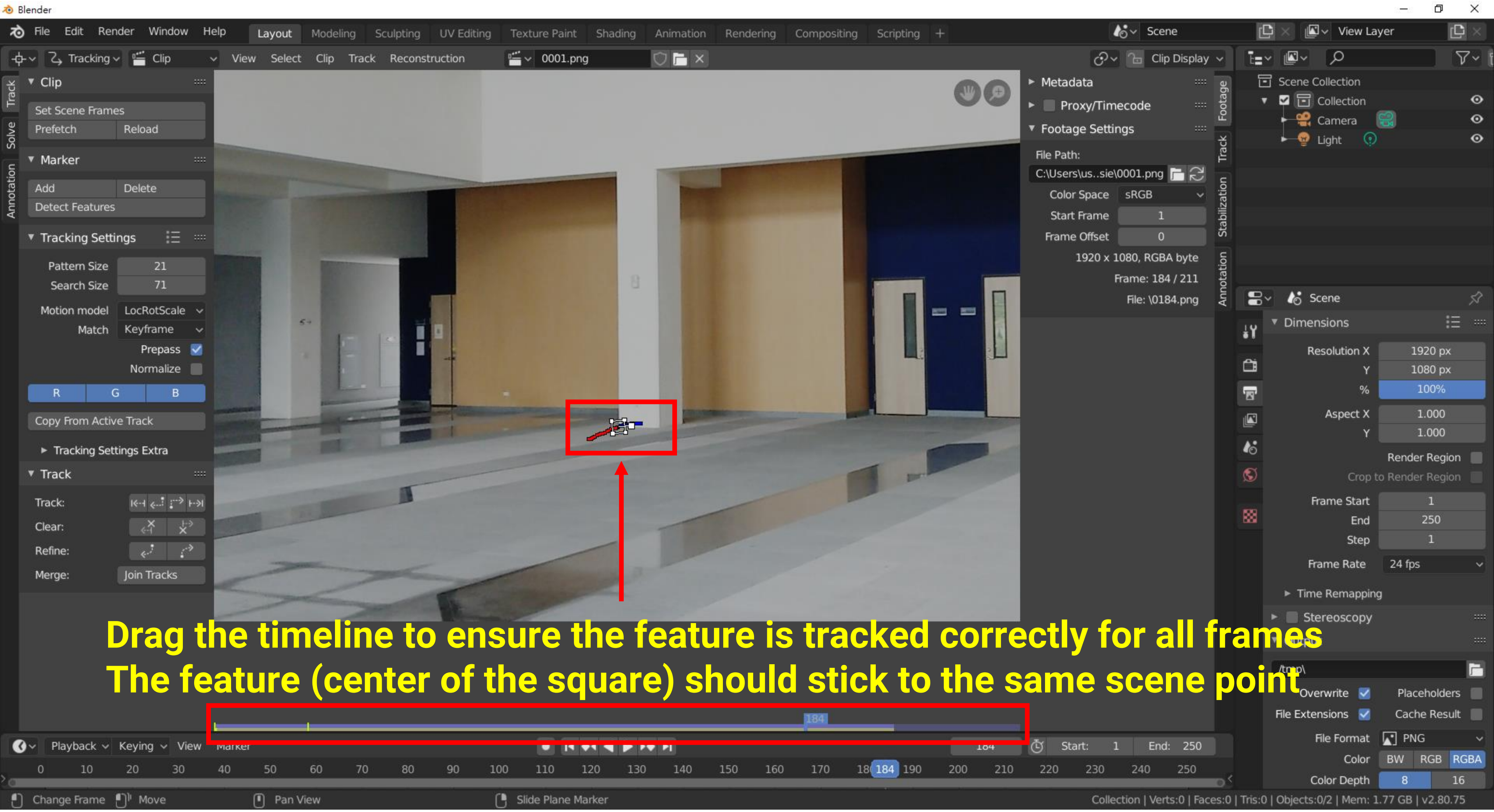
Feature Detection (2)



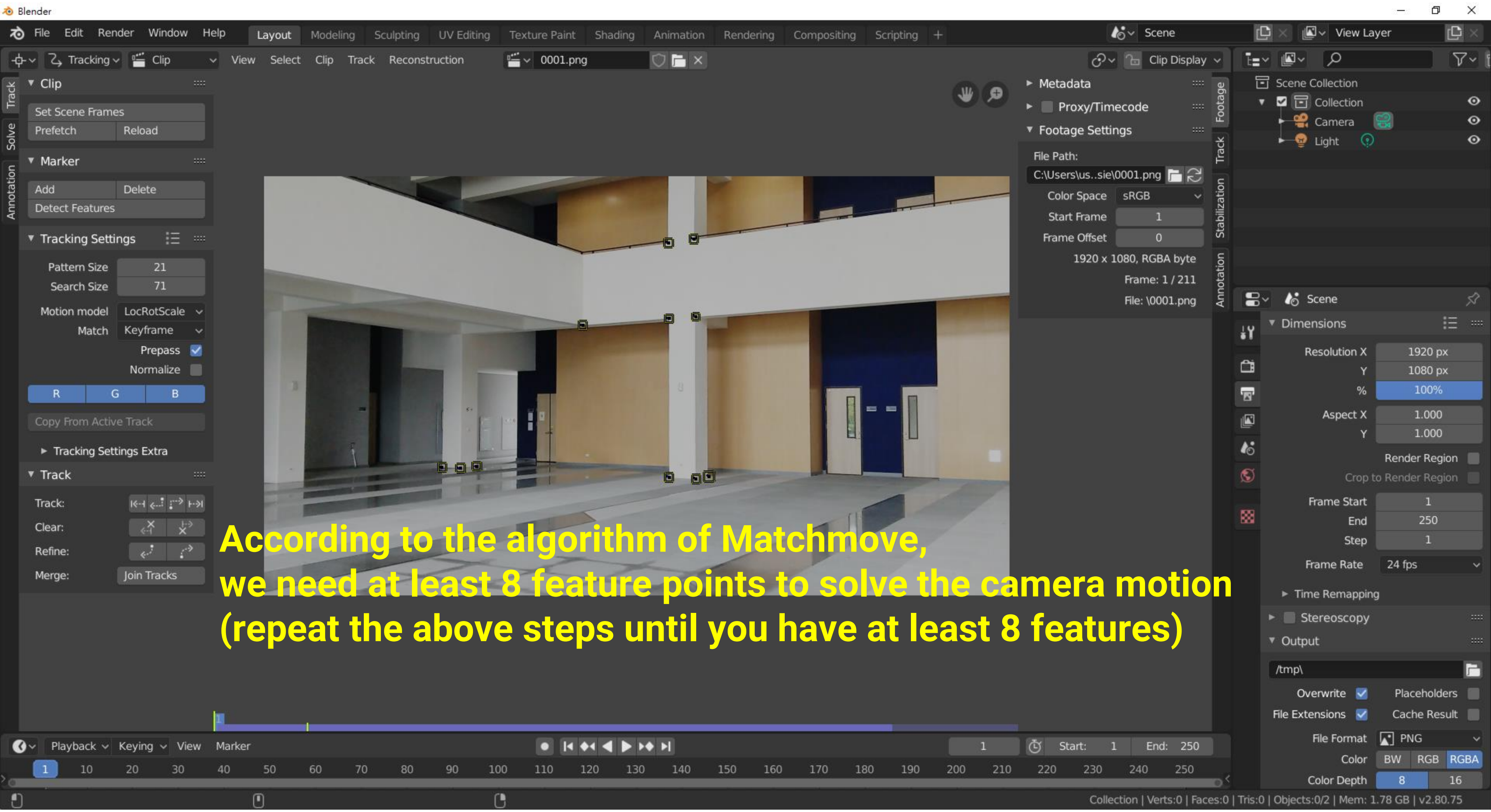
Feature Detection (3)



Feature Detection (4)

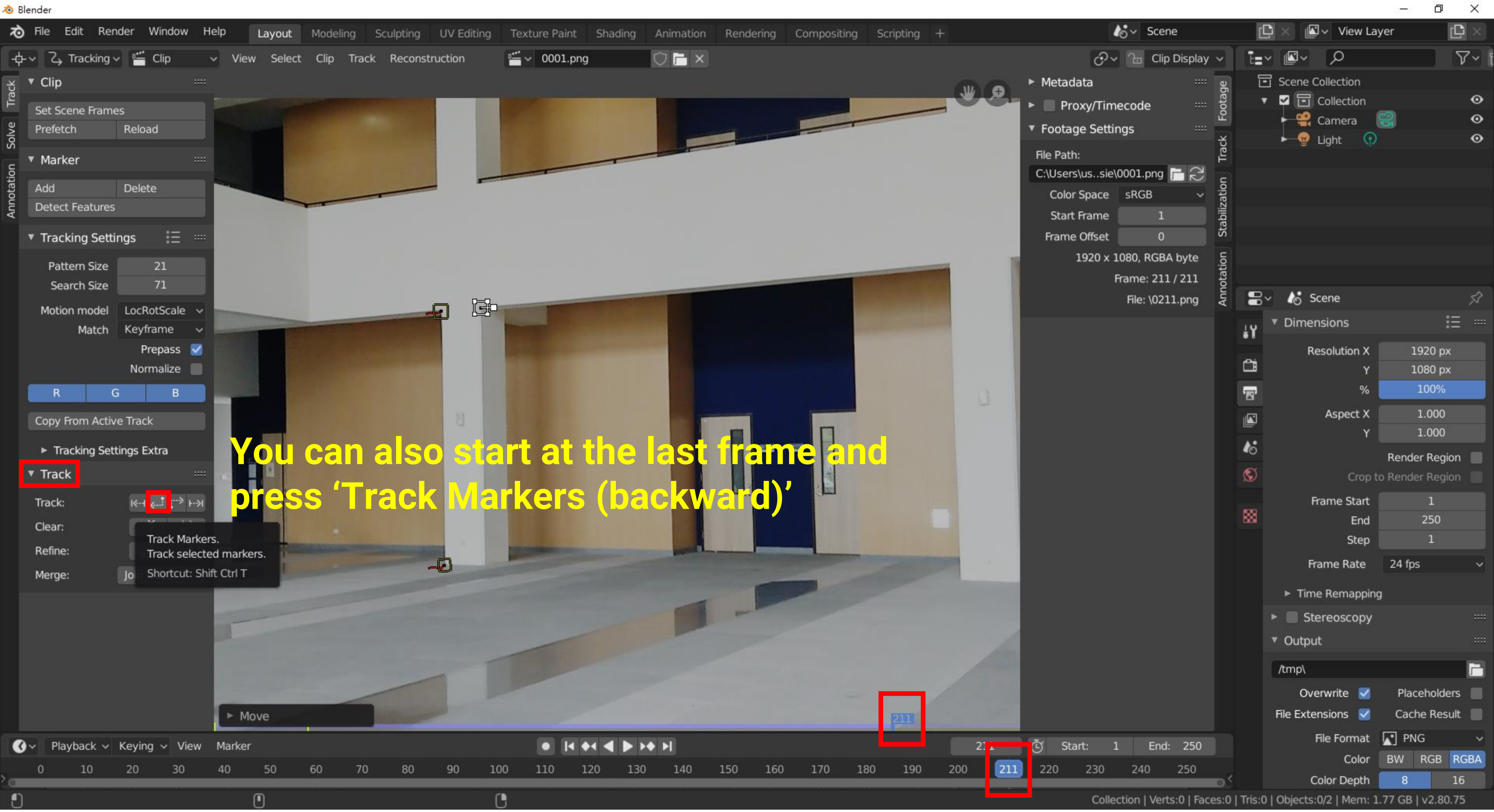


Feature Detection (5)

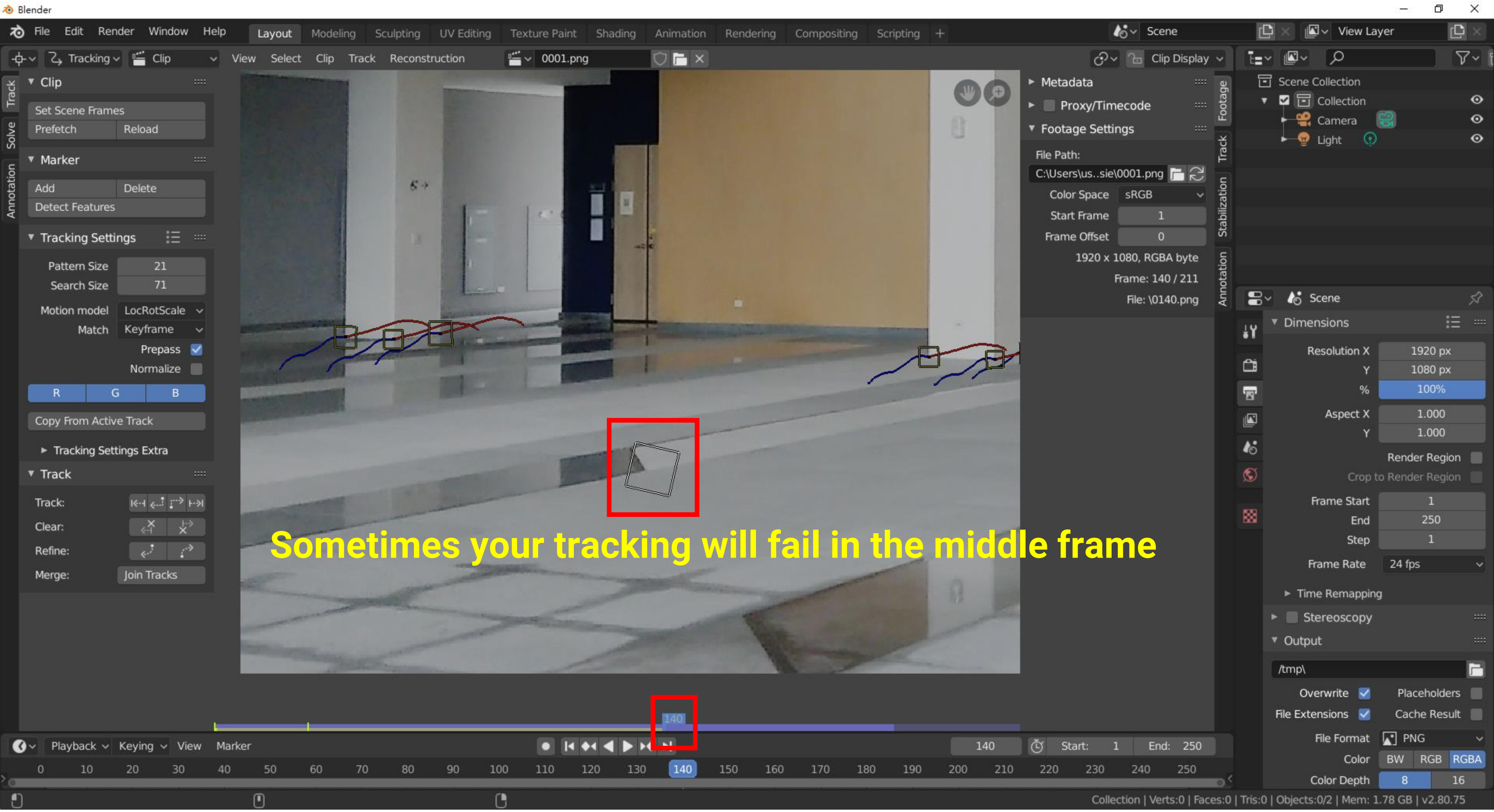


According to the algorithm of Matchmove,
we need at least 8 feature points to solve the camera motion
(repeat the above steps until you have at least 8 features)

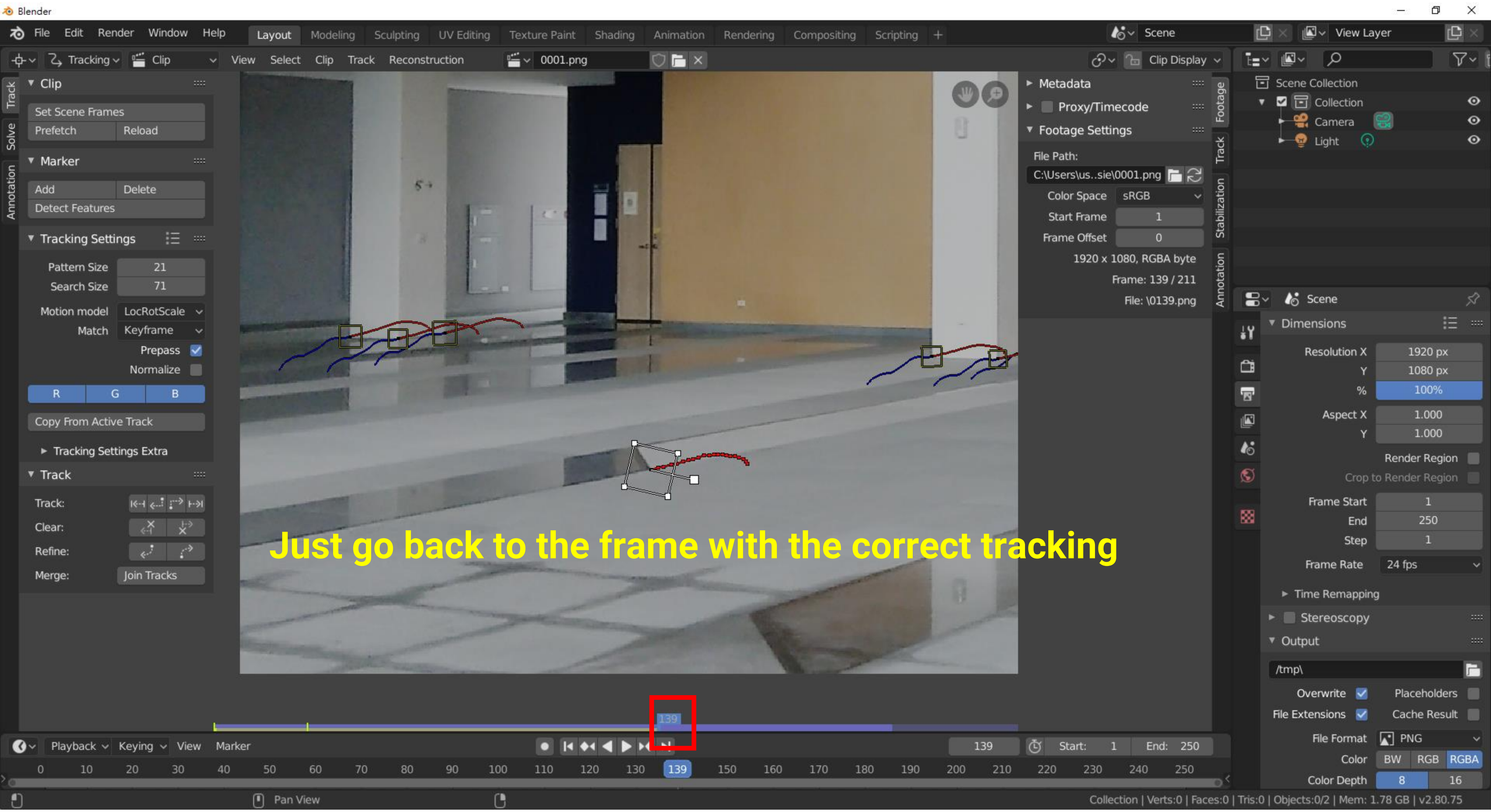
Feature Detection (6)



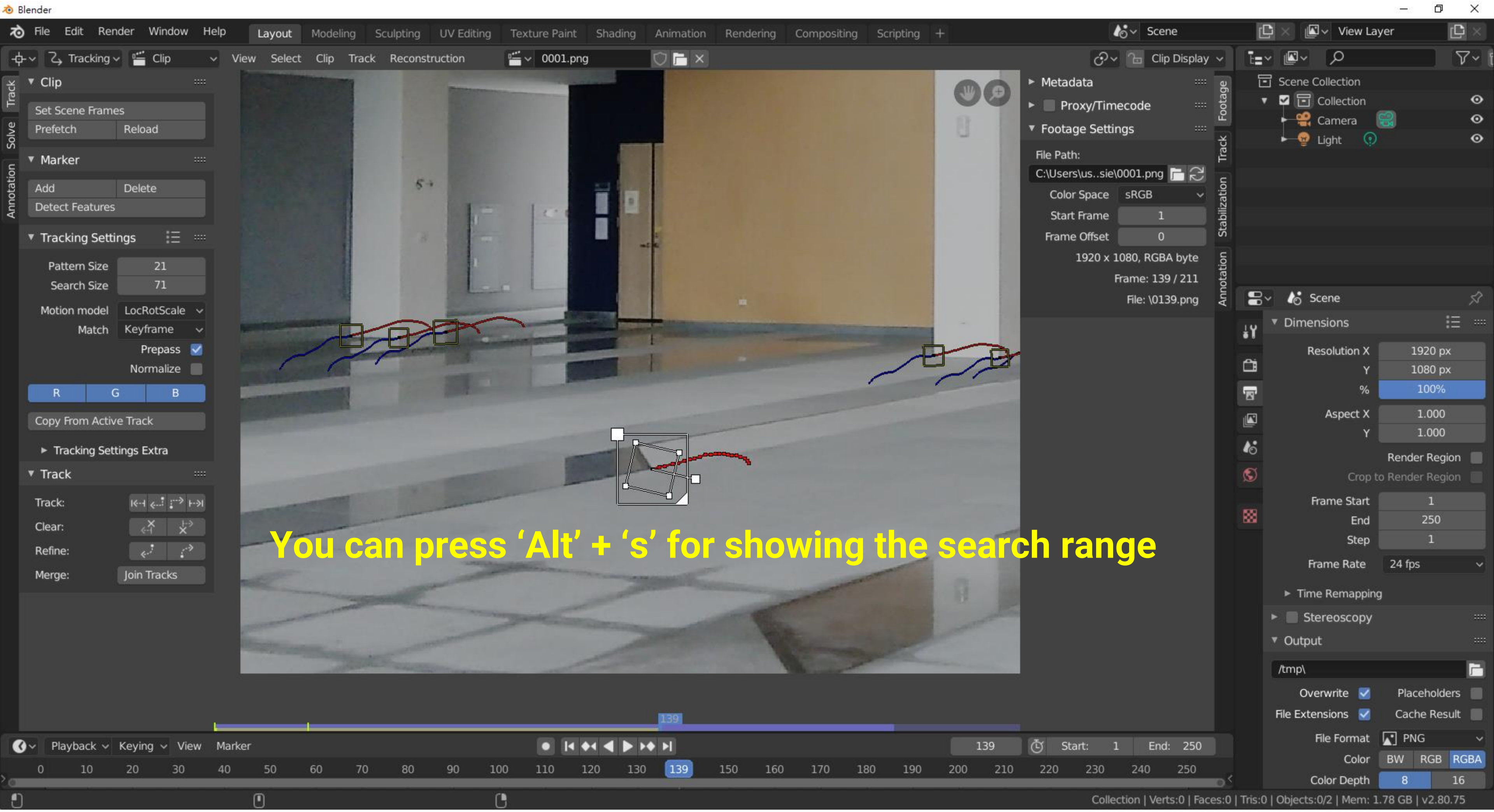
Feature Detection (7)



Feature Detection (8)

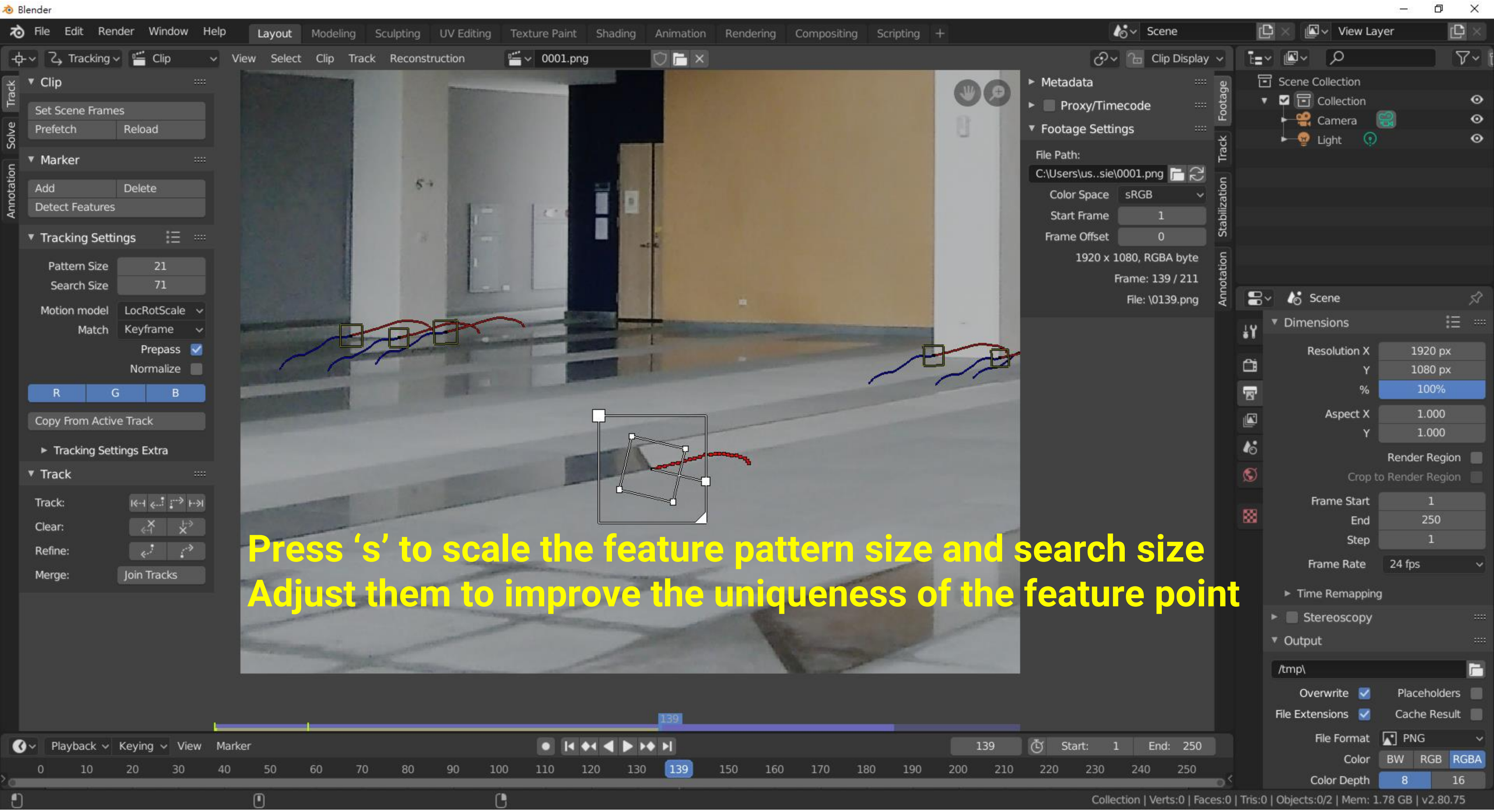


Feature Detection (9)



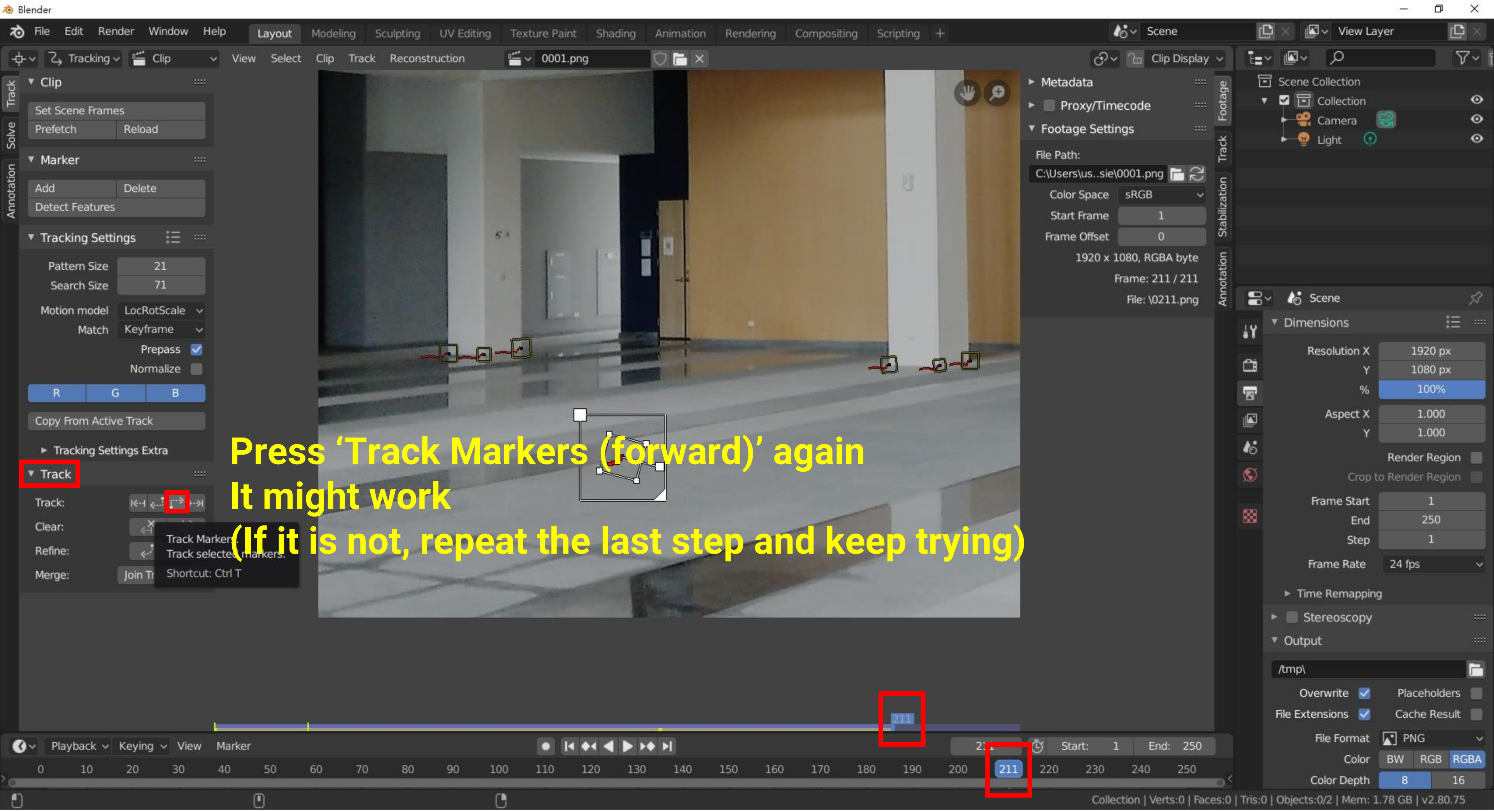
You can press ‘Alt’ + ‘s’ for showing the search range

Feature Detection (10)



Press 's' to scale the feature pattern size and search size
Adjust them to improve the uniqueness of the feature point

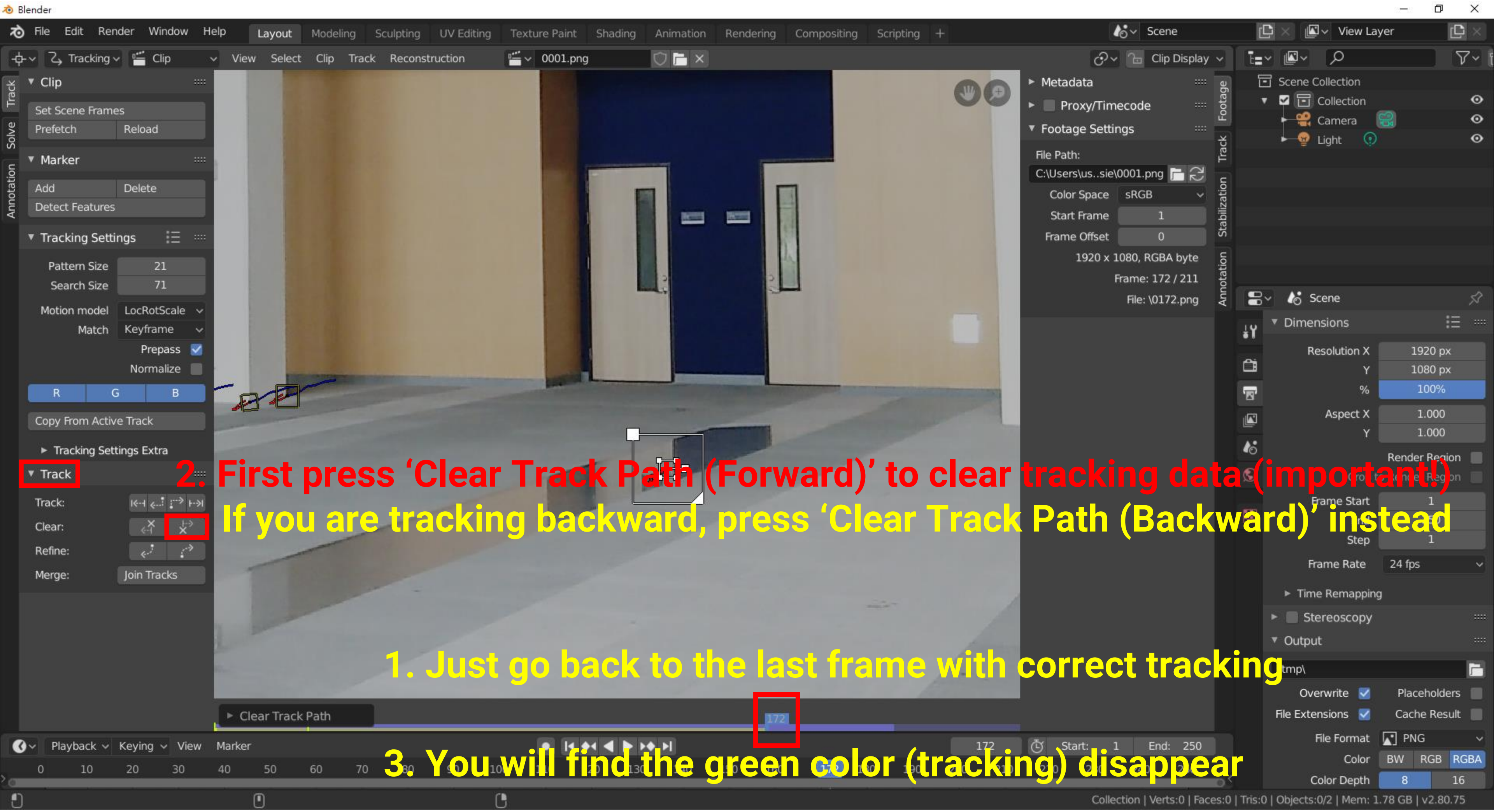
Feature Detection (11)



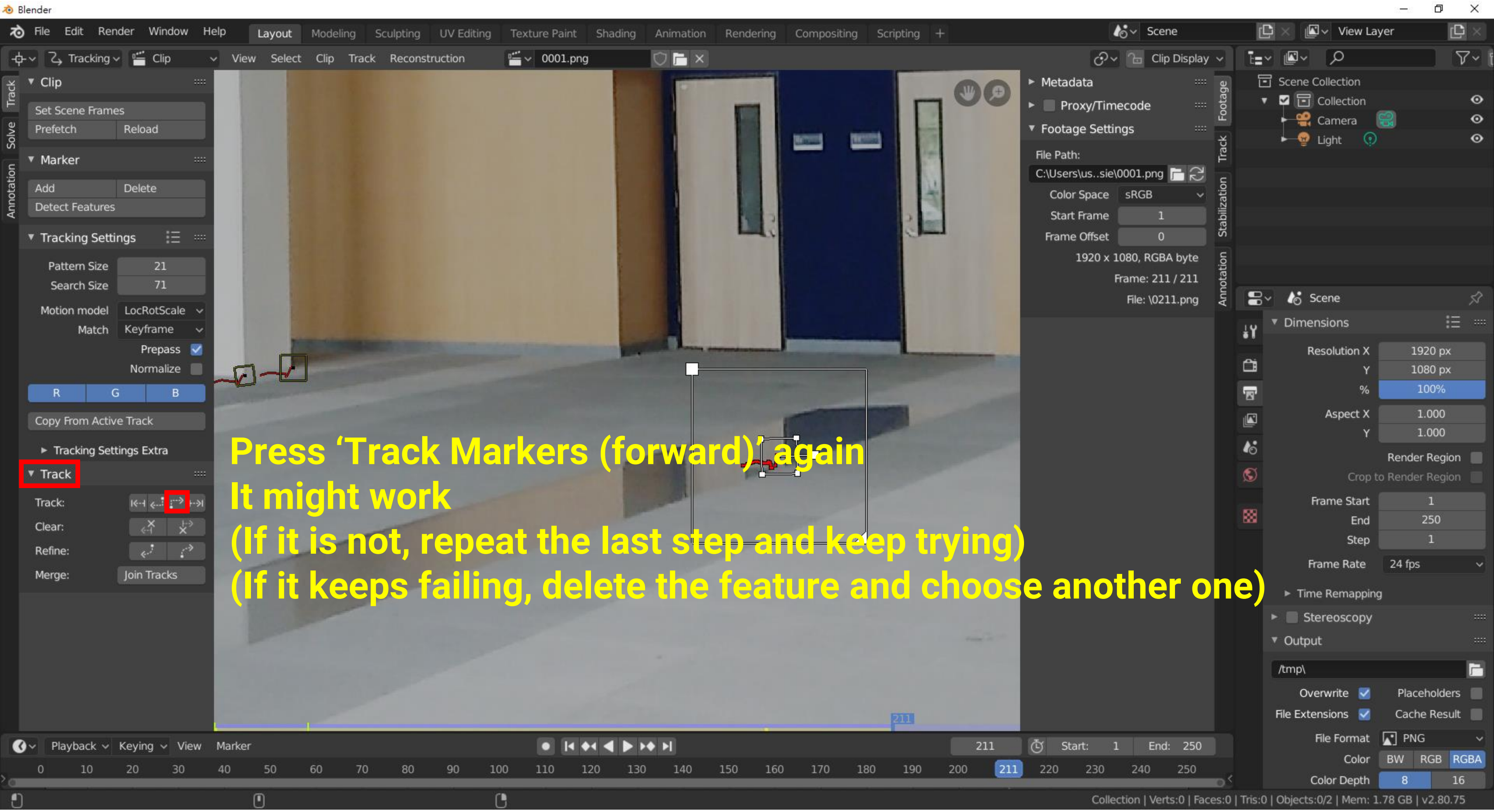
Press ‘Track Markers (forward)’ again
It might work
(If it is not, repeat the last step and keep trying)

Feature Detection (12)

Feature Detection (13)

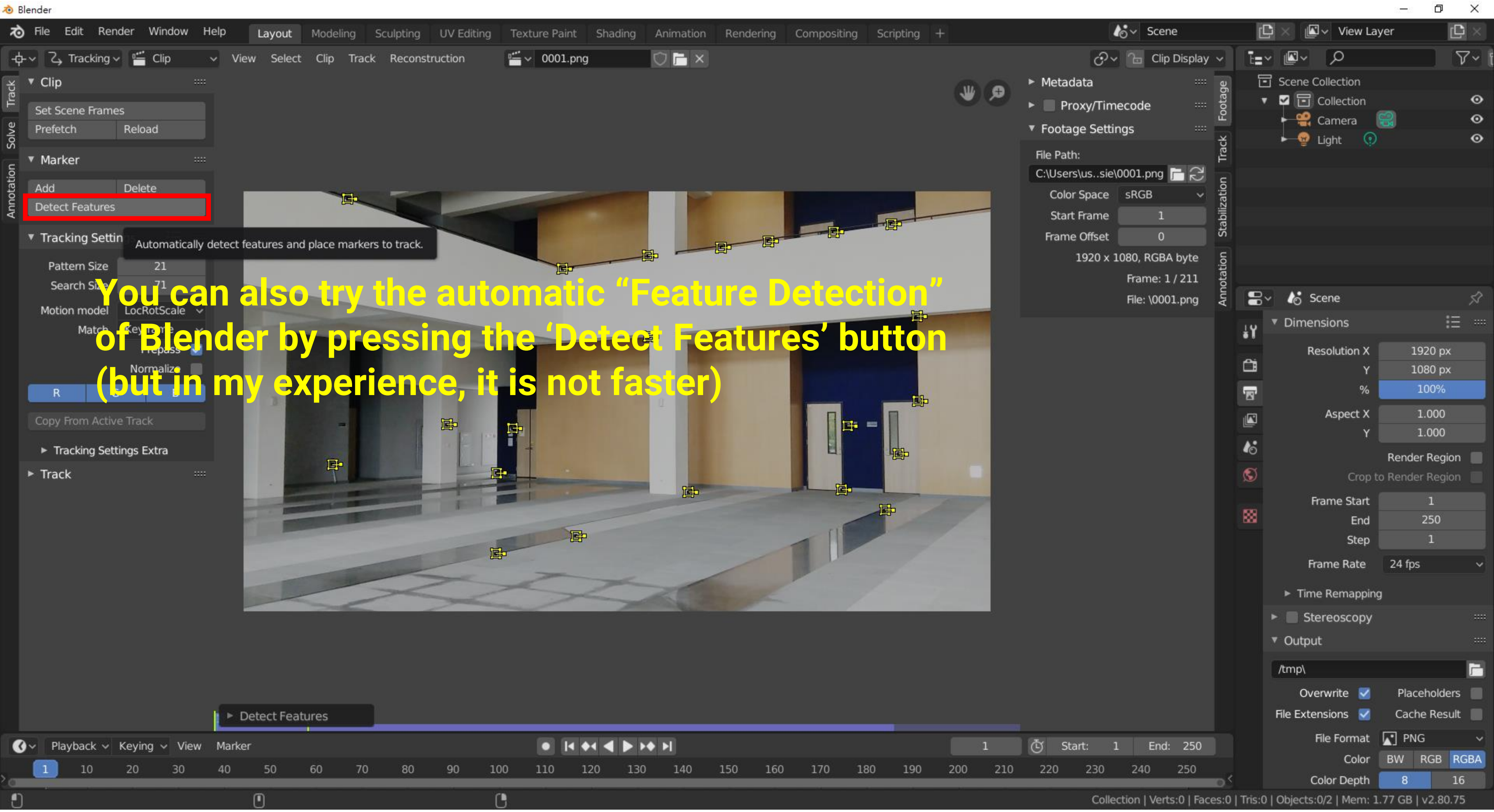


Feature Detection (14)



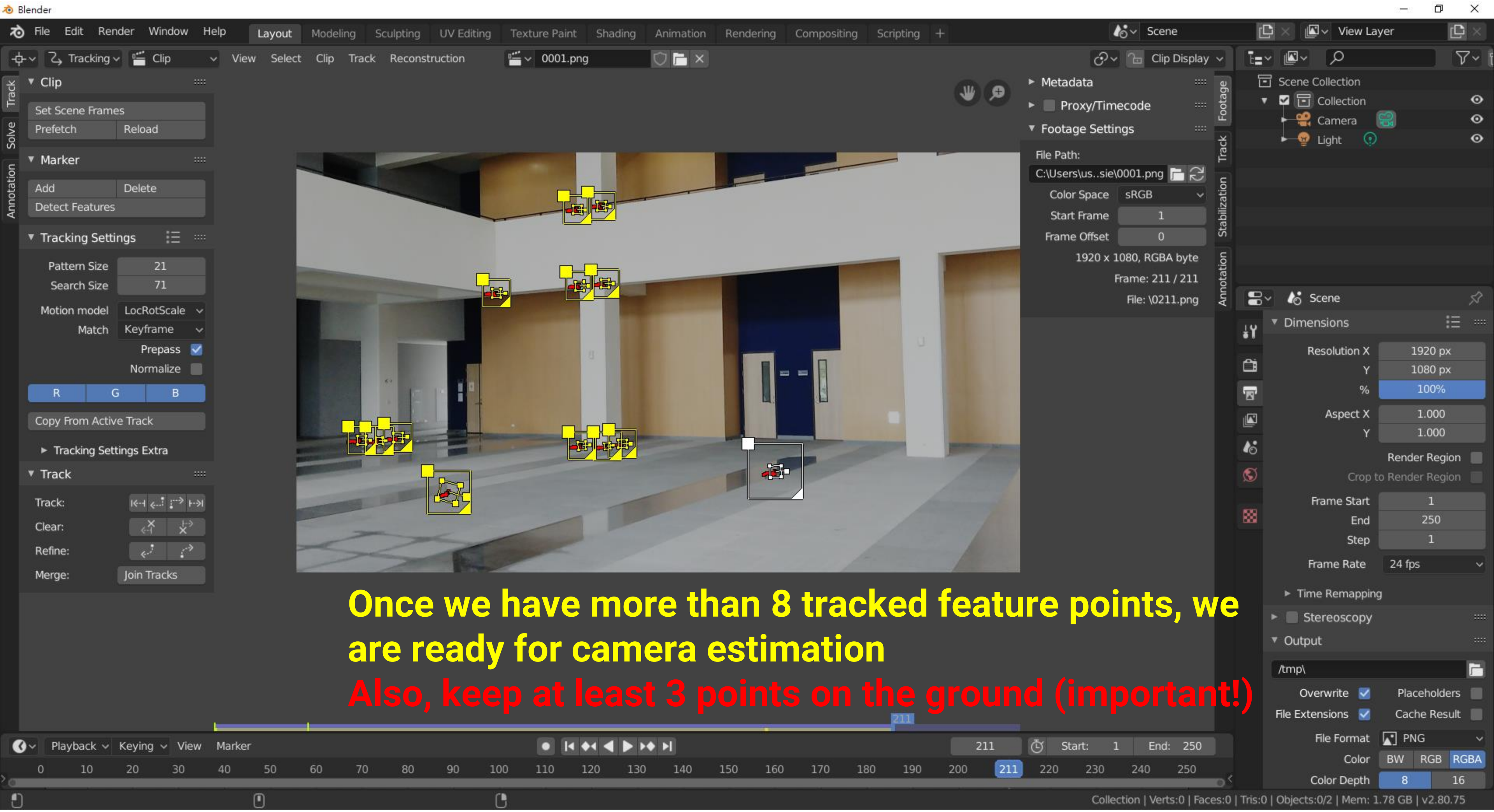
Press 'Track Markers (forward)' again
It might work
(If it is not, repeat the last step and keep trying)
(If it keeps failing, delete the feature and choose another one)

Feature Detection (15)



You can also try the automatic "Feature Detection" of Blender by pressing the 'Detect Features' button (but in my experience, it is not faster)

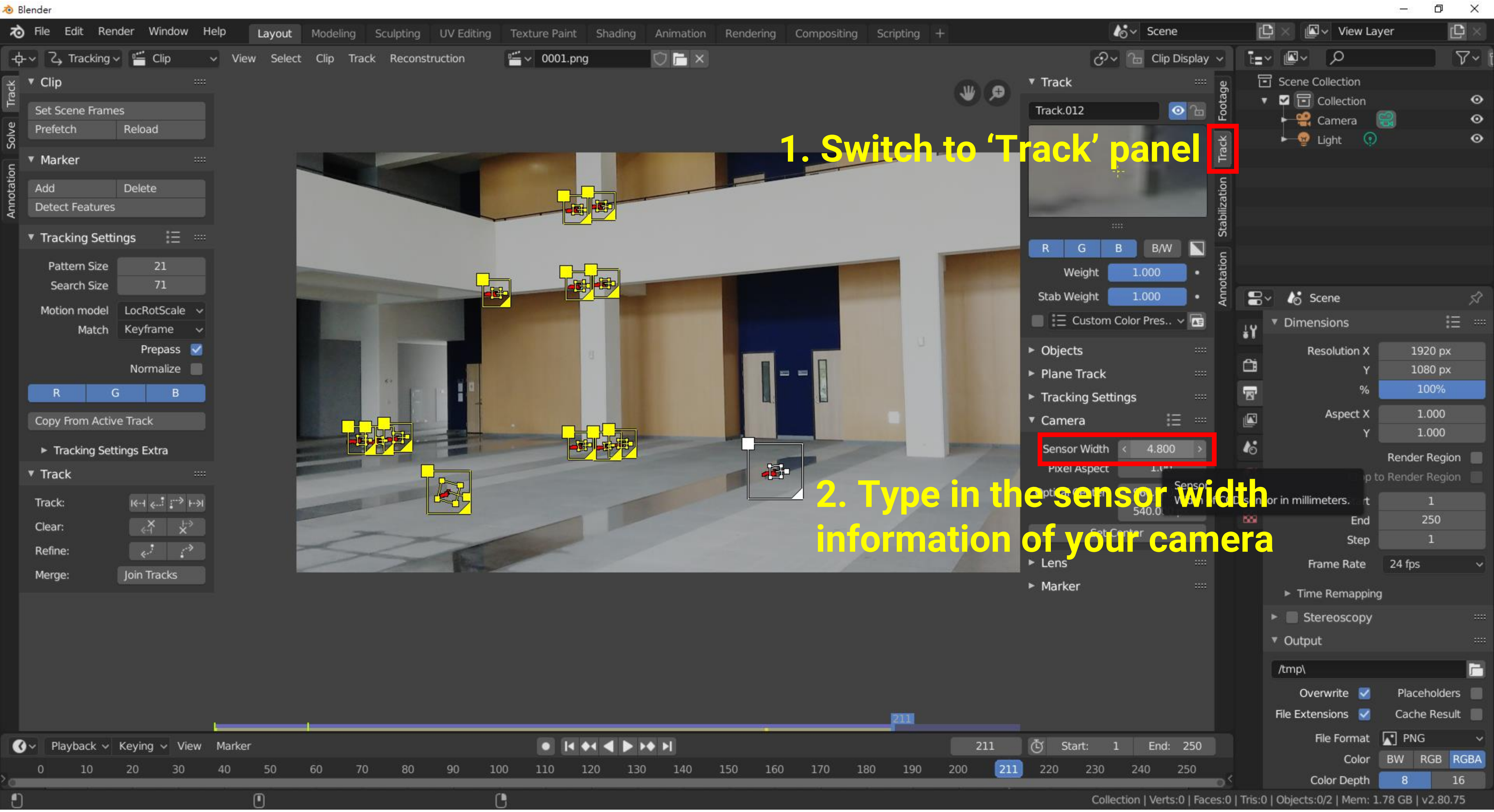
Feature Detection (16)



Once we have more than 8 tracked feature points, we are ready for camera estimation

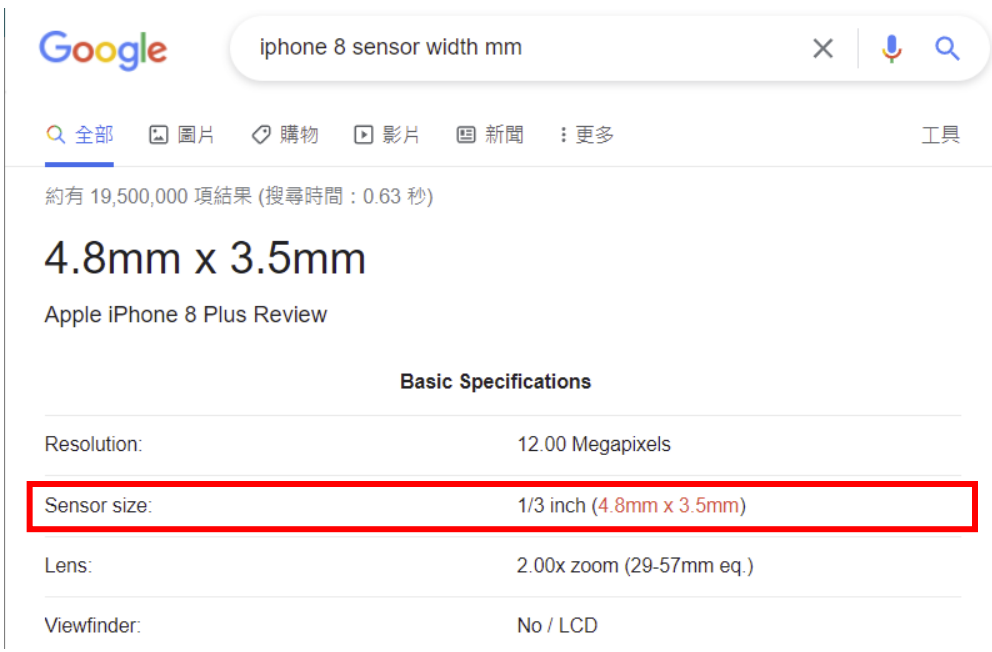
Also, keep at least 3 points on the ground (important!)

Setting Camera Parameters (1)



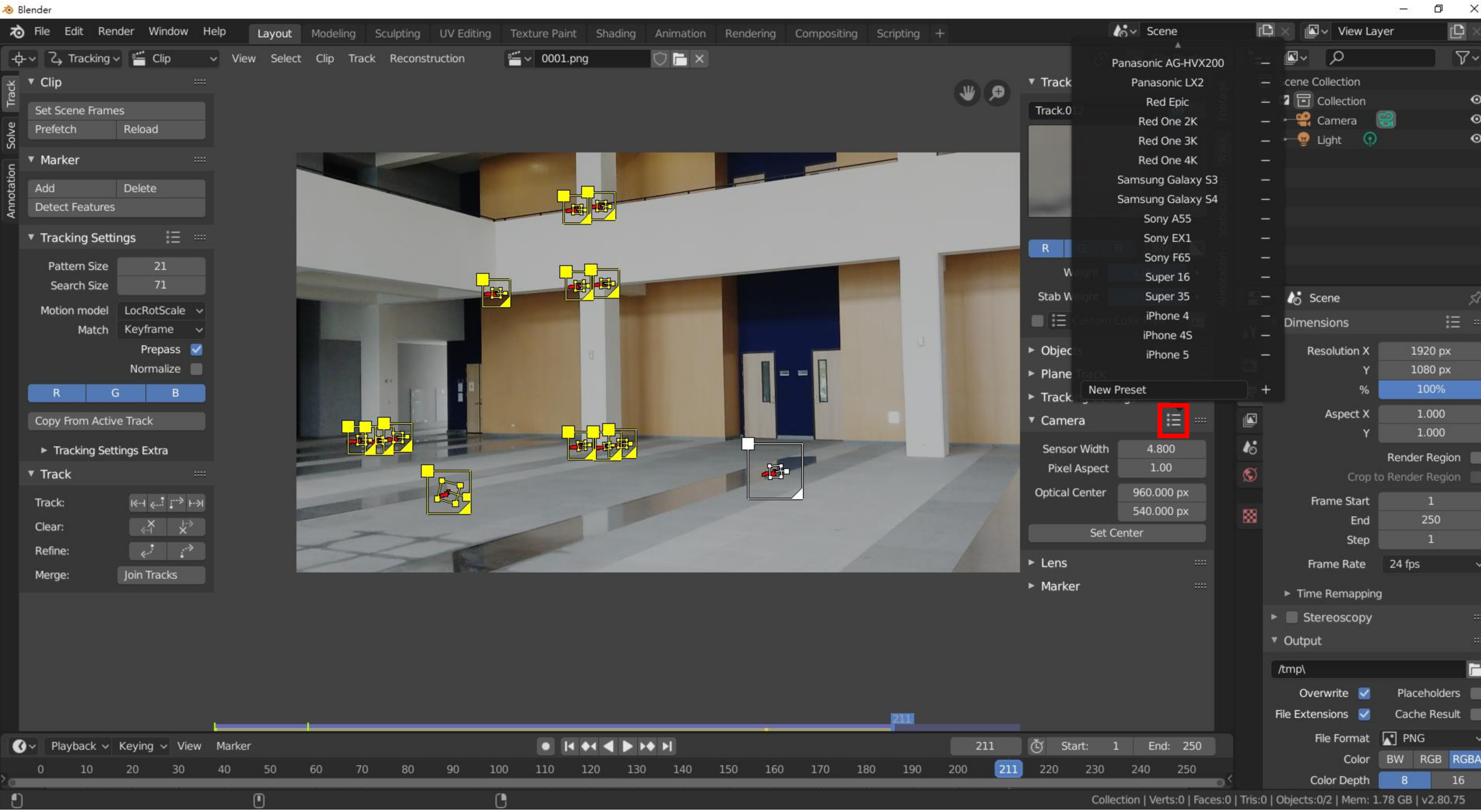
Camera Parameters

- **How do we know the camera parameters?**
  - Google it on the internet
    - https://www.photocounter.com.au/wp-content/uploads/2013/01/sensor-size-table.pdf
  - You can also use Blender’s preset

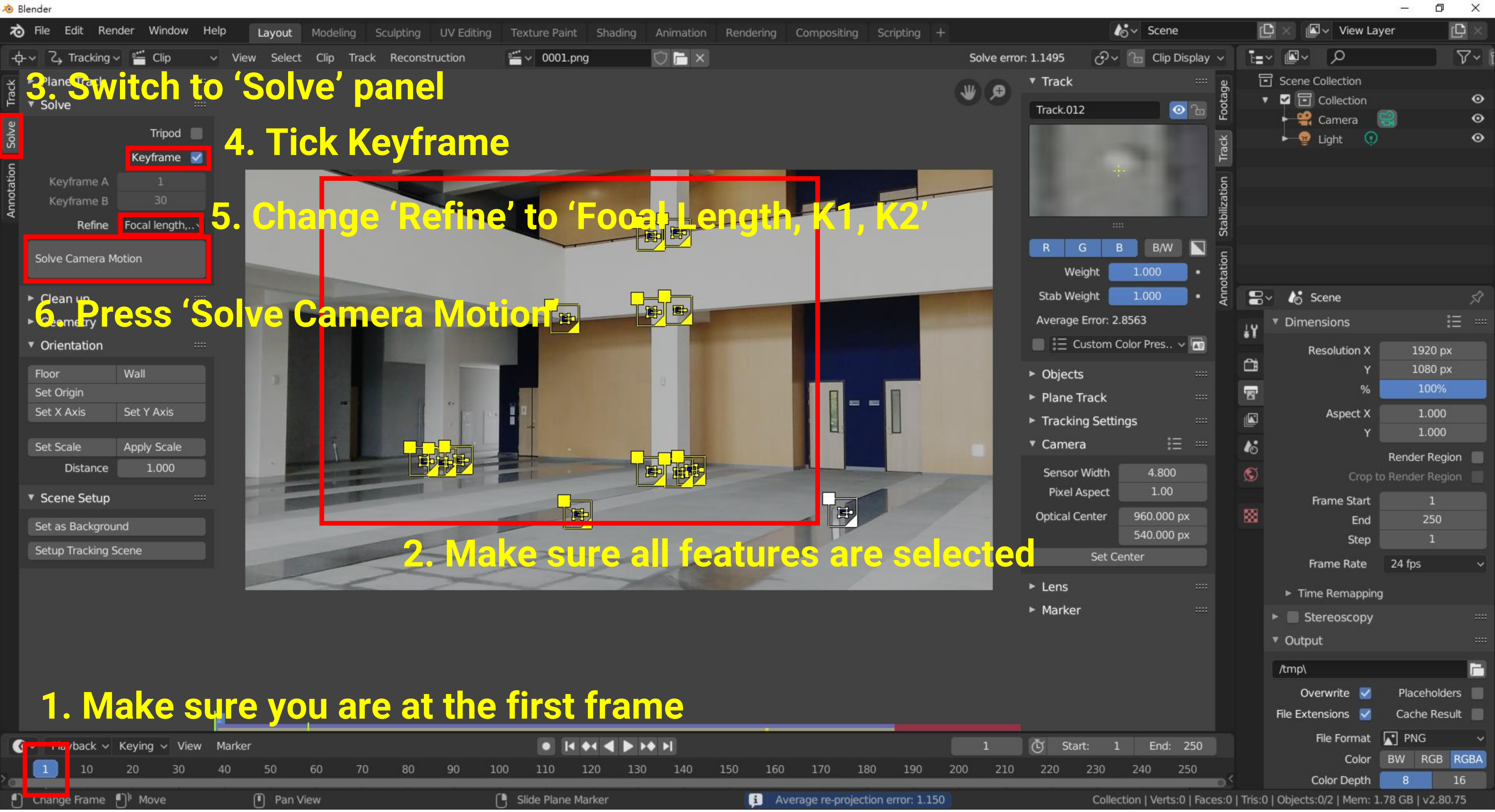


Sensor “Type”	Imaging Area Dimensions			
	Diagonal (mm)	Width (mm)	Height (mm)	Area (mm^2)
1/6"	2.7	2.46	1.8	4.43
1/4"	4.5	3.6	2.7	9.72
1/3.6"	5.0	4.0	3.0	12.0
1/3.2"	5.68	4.54	3.42	15.53
1/3"	6.0	4.8	3.6	17.28
1/2.7"	6.72	5.37	4.04	21.69
1/2.5"	7.18	5.76	4.29	24.71
1/2.4"	7.66	5.92	4.57	27.05
1/2.33"	7.7	6.12	4.51	27.60
1/2.3"	7.8	6.17	4.55	28.07
1/2"	8.0	6.4	4.8	30.72
1/1.8"	8.93	7.18	5.32	38.20
1/1.75"	9.23	7.38	5.54	40.89
1/1.72”	9.25	7.40	5.55	41.07
1/1.7"	9.5	7.6	5.7	43.32
1/1.6"	10.07	8.08	6.01	48.56
2/3"	11.07	8.8	6.6	58.08
1"	16.0	12.8	9.6	122.88
4/3"	22.5	17.3	13.0	243.00

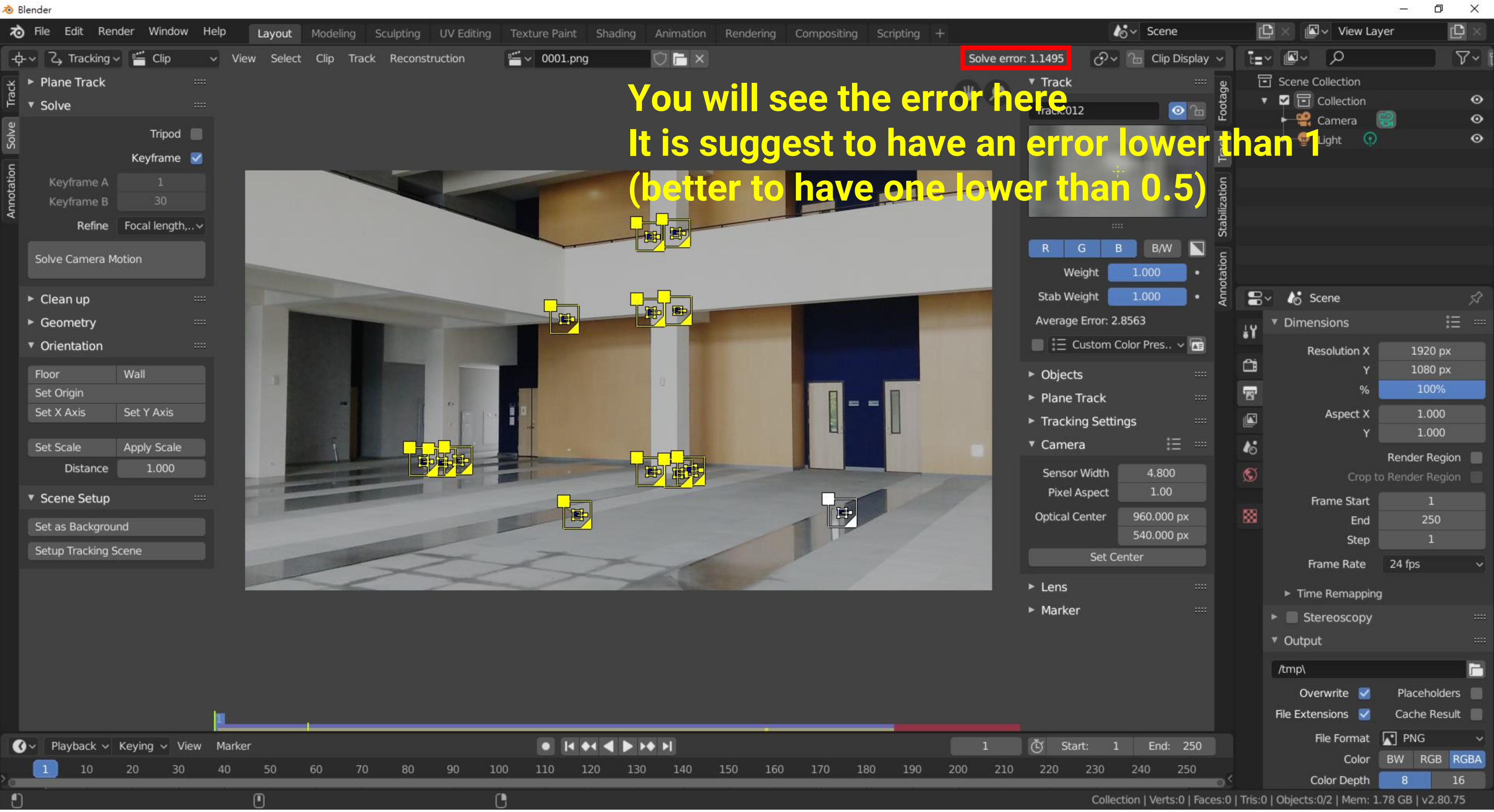
Setting Camera Parameters (2)



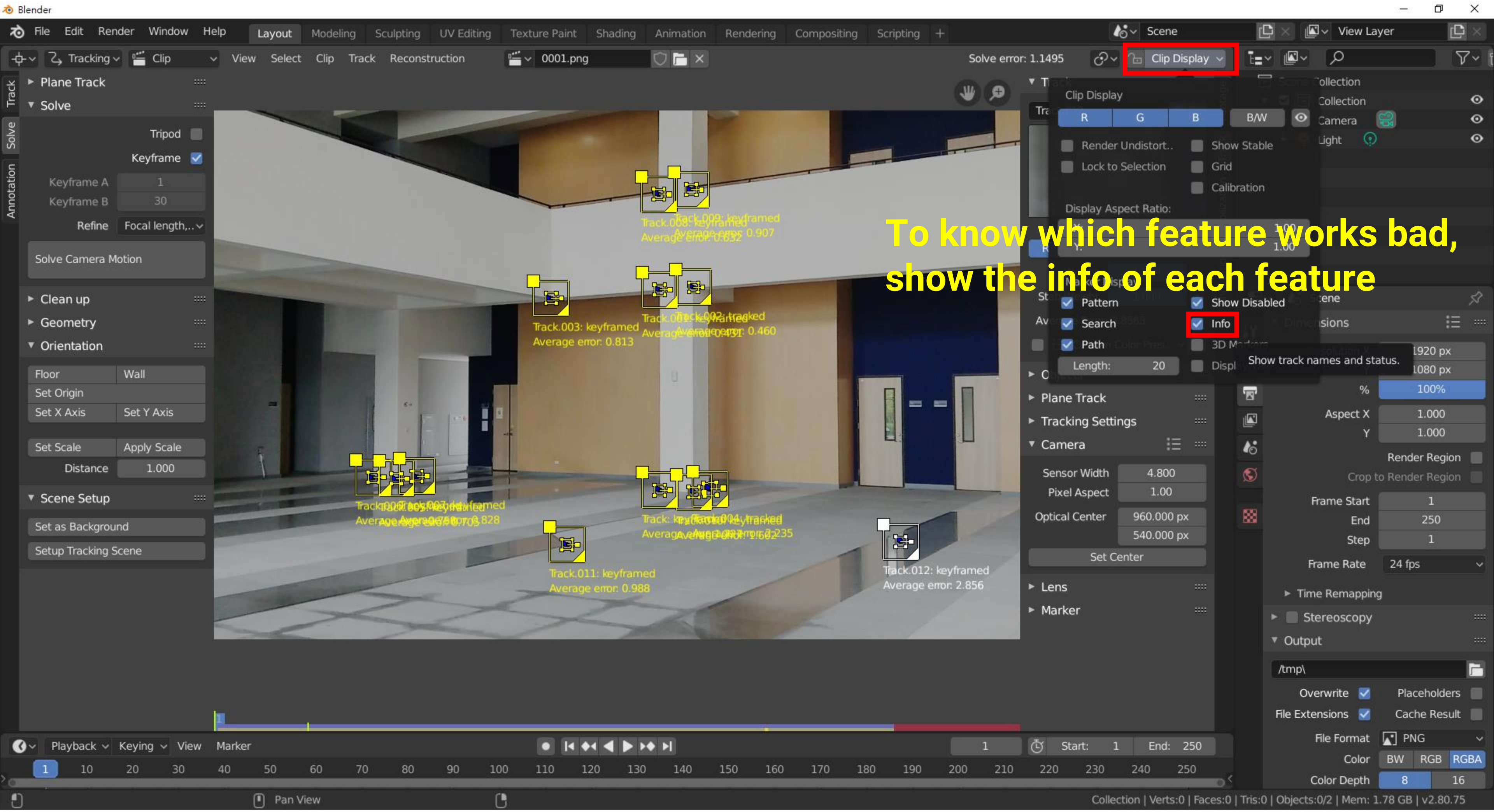
Solving Camera Motion (1)



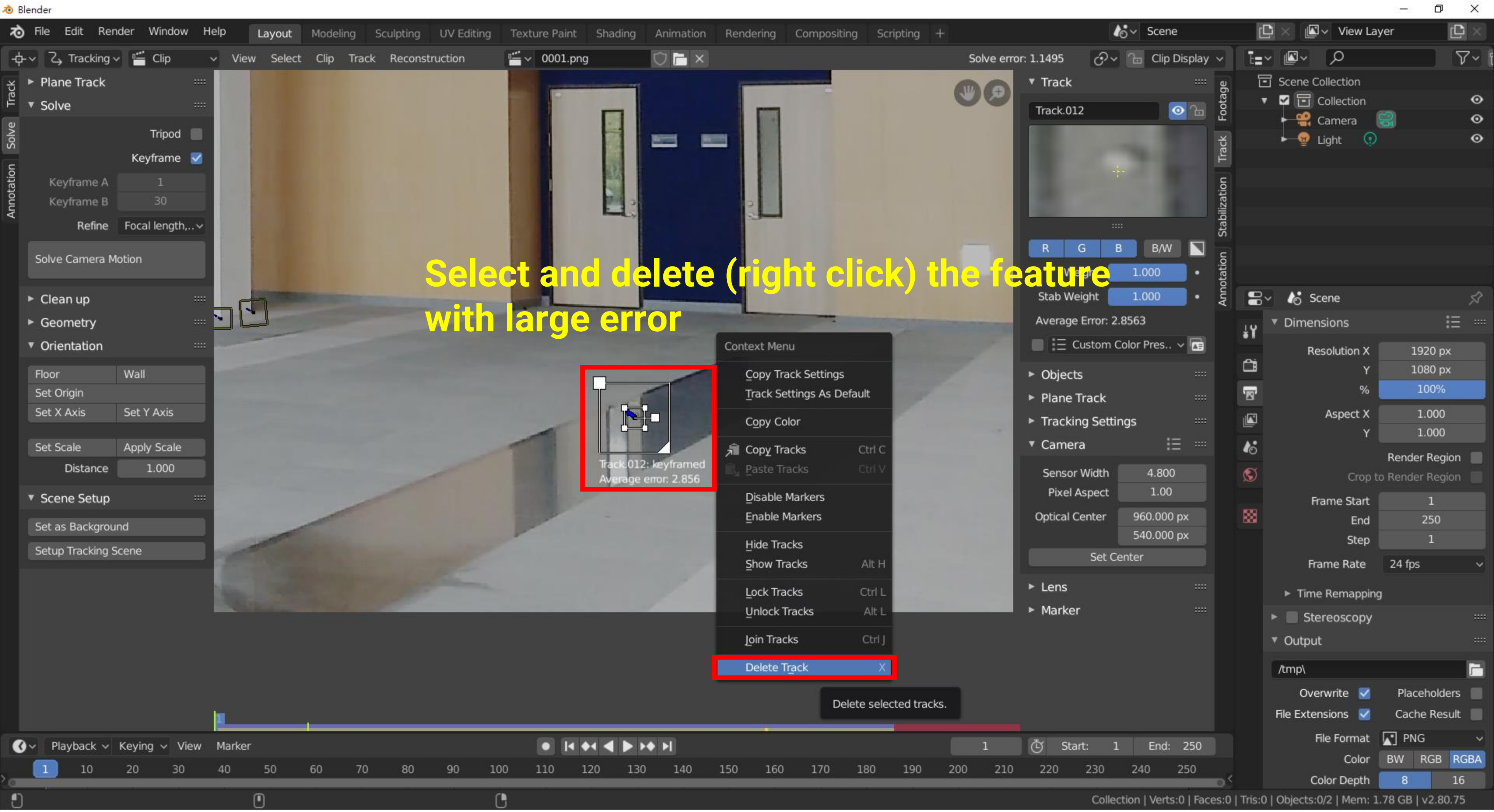
Solving Camera Motion (2)



Solving Camera Motion (3)

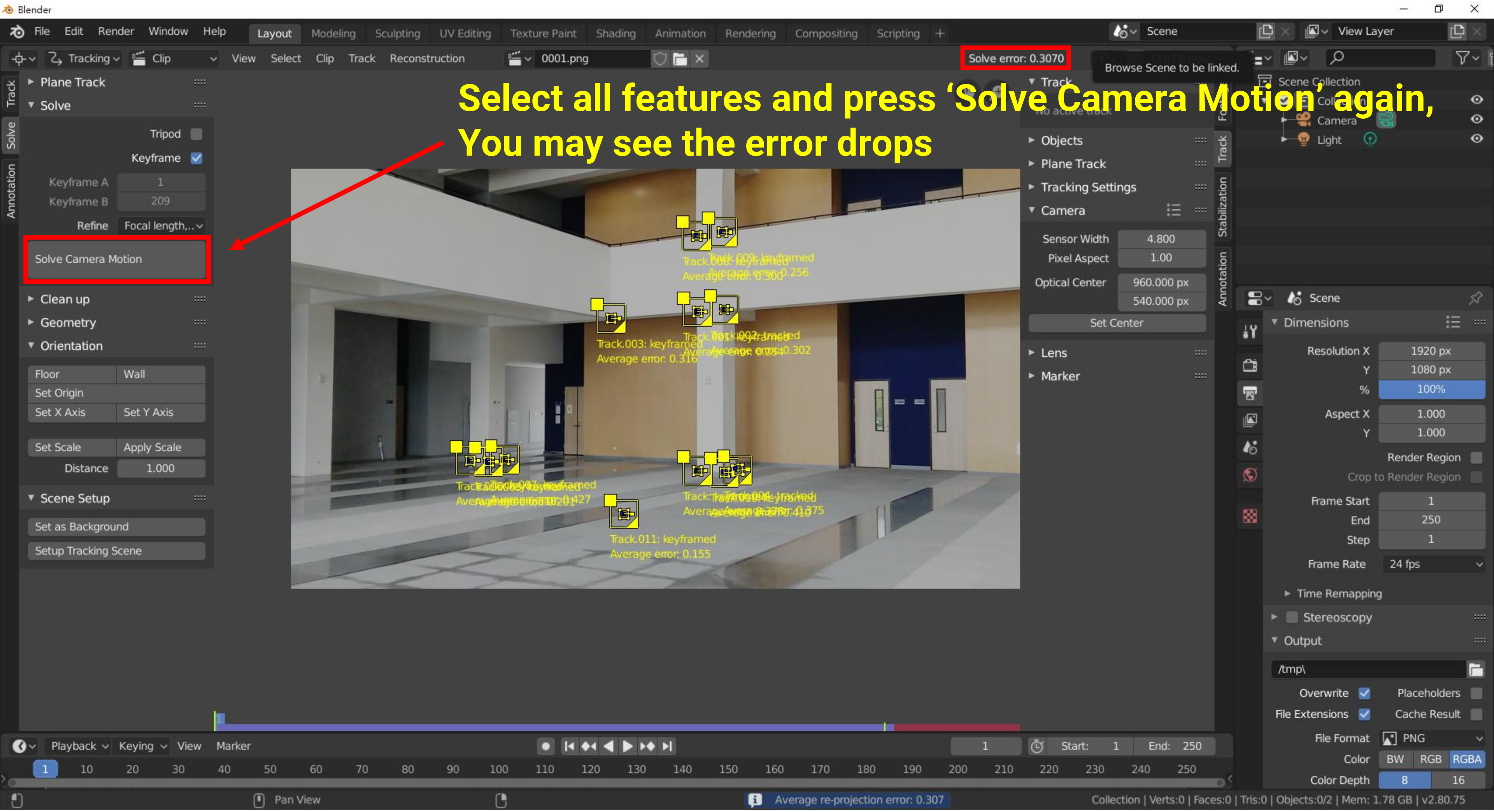


Solving Camera Motion (4)

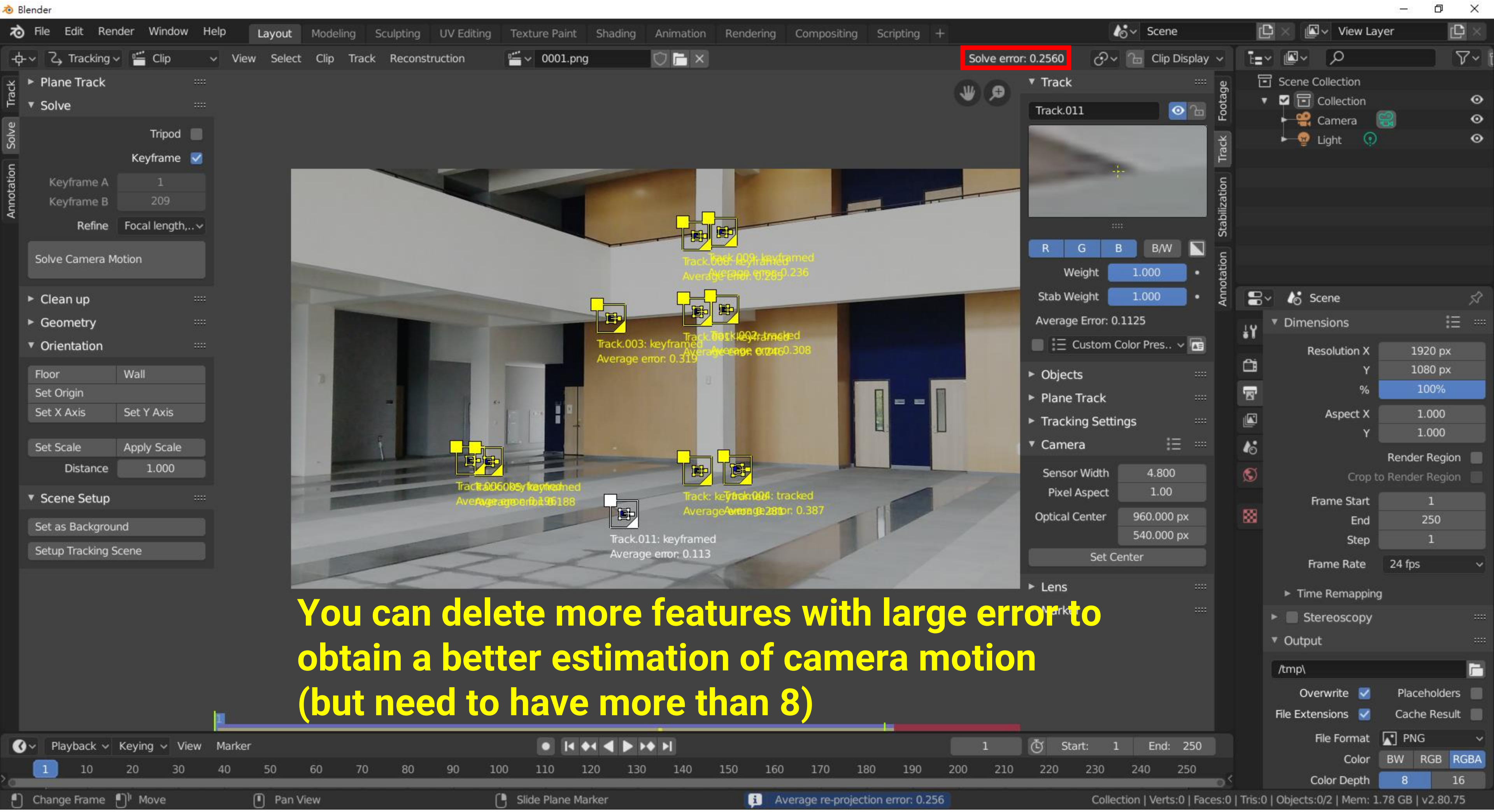


Solving Camera Motion (5)

Select all features and press ‘Solve Camera Motion’ again,
You may see the error drops

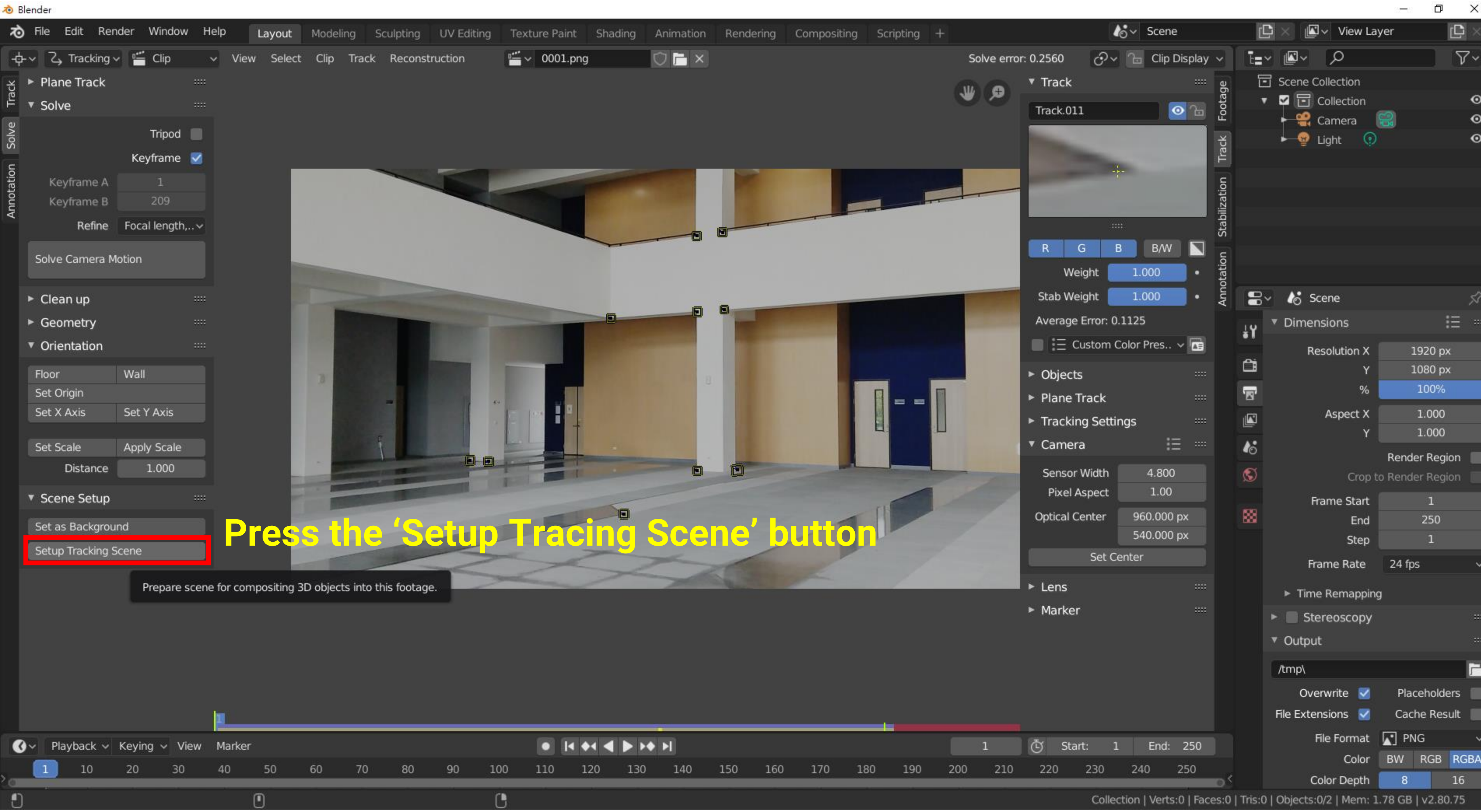


Solving Camera Motion (6)

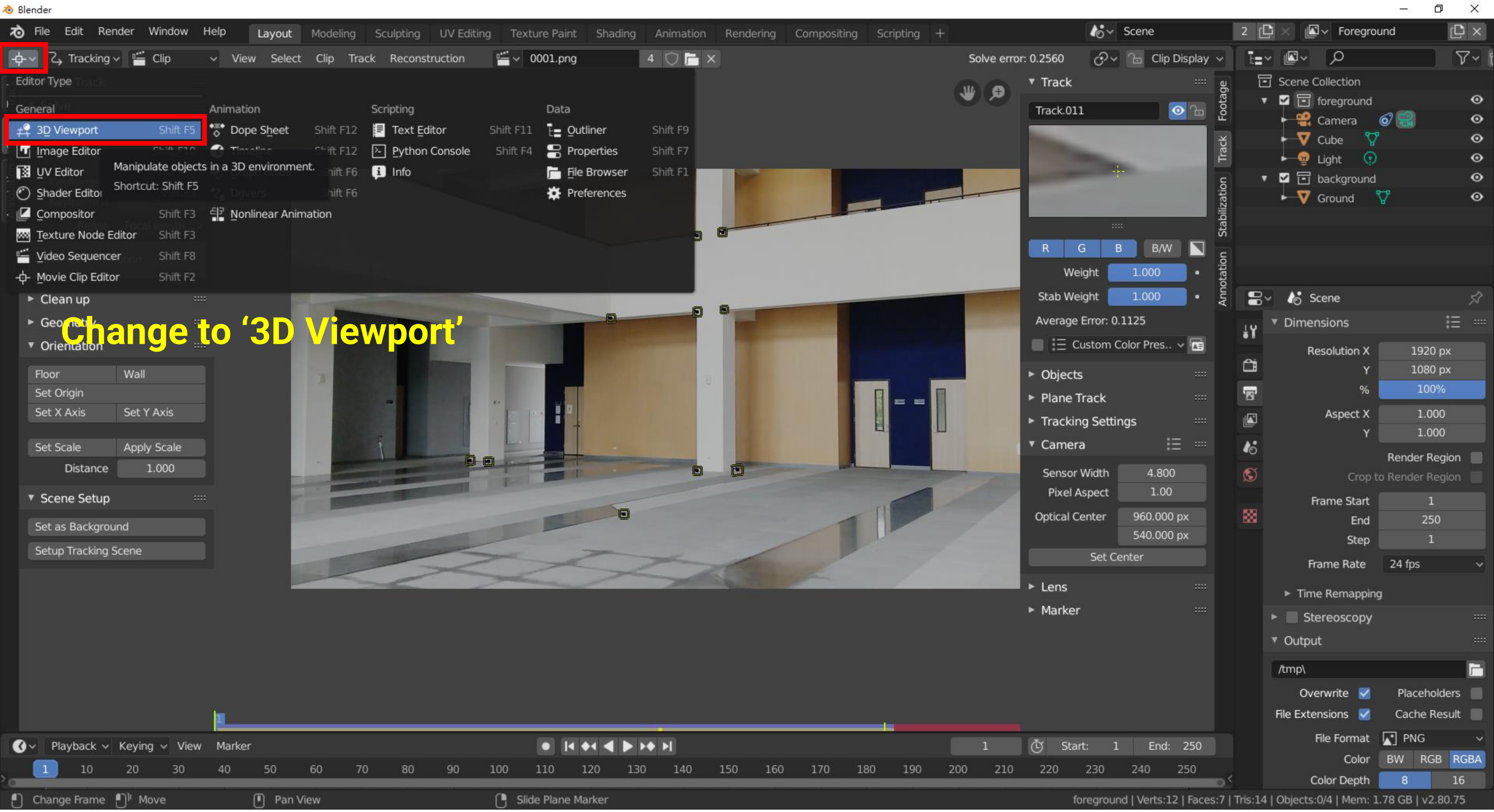


You can delete more features with large error to obtain a better estimation of camera motion (but need to have more than 8)

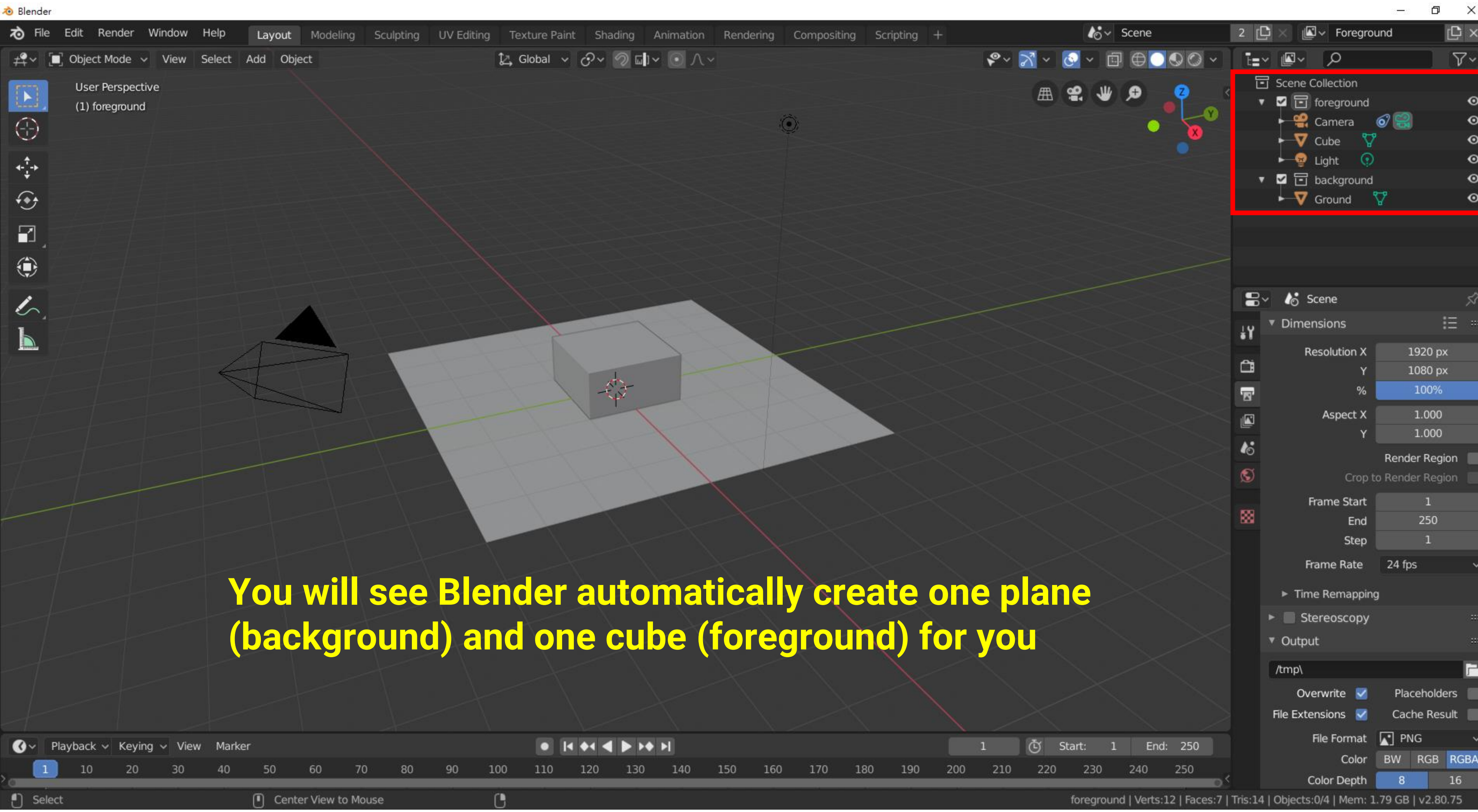
Setup Tracking Scene (1)



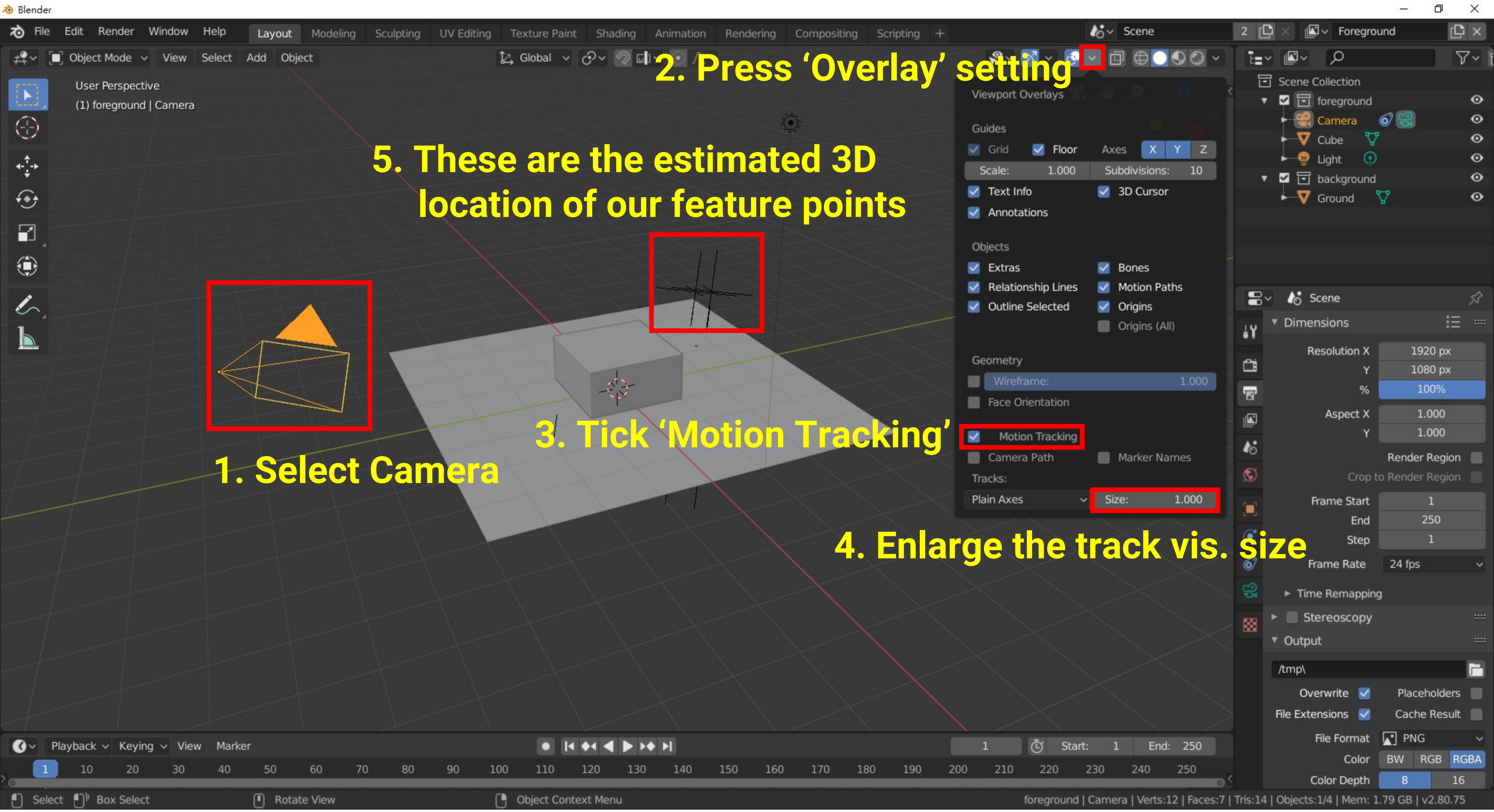
Setup Tracking Scene (2)



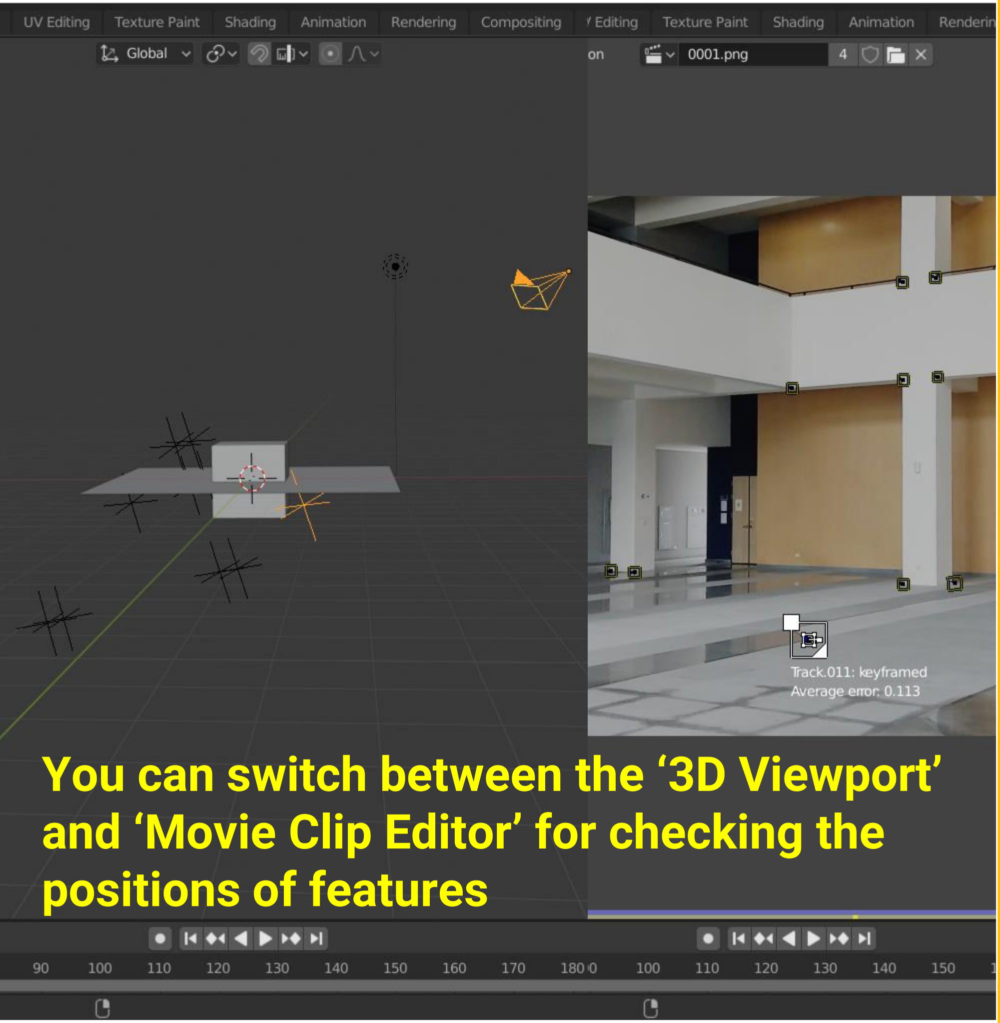
Setup Tracking Scene (3)



Setup Tracking Scene (4)

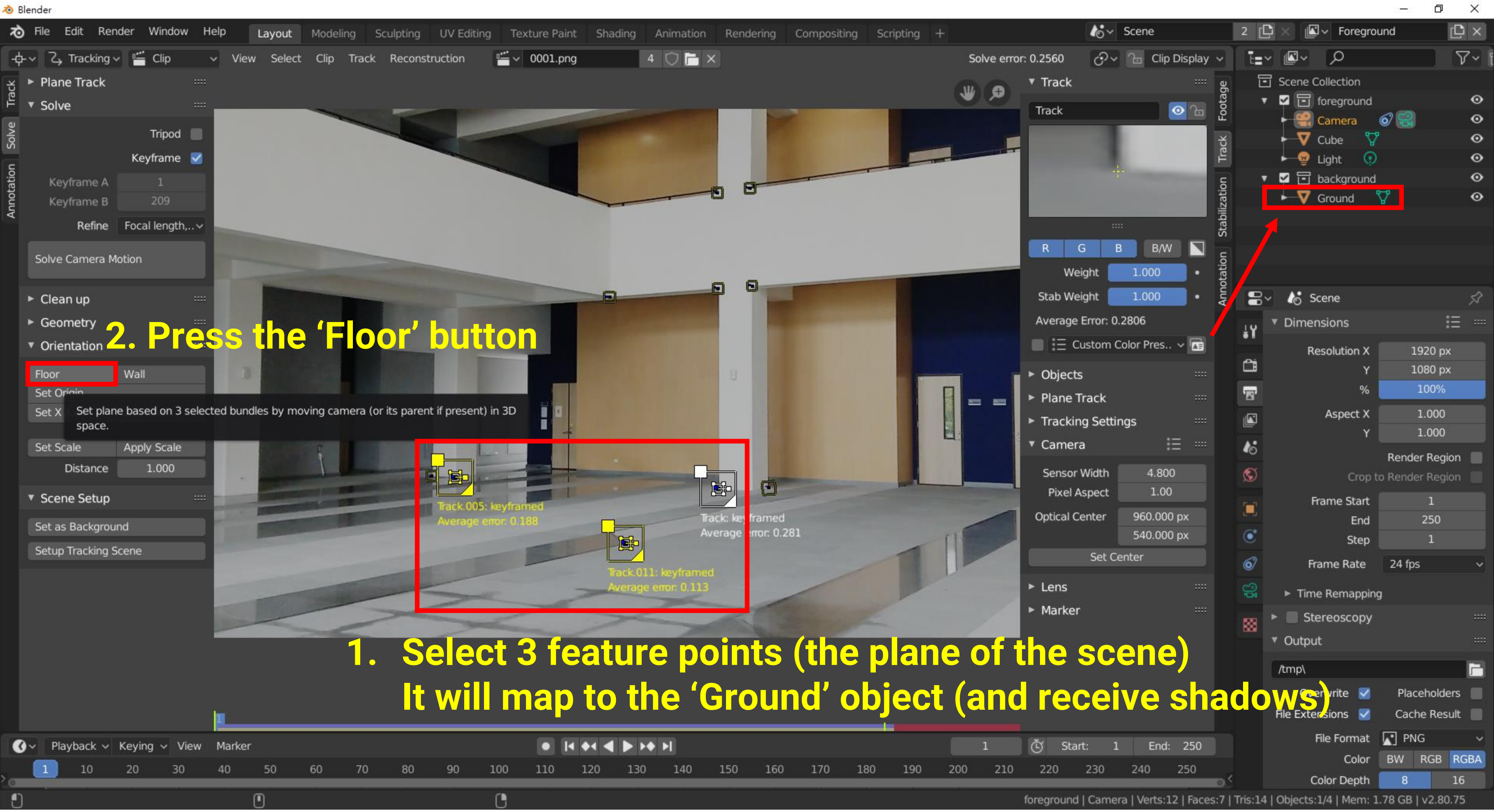


Setup Tracking Scene (5)

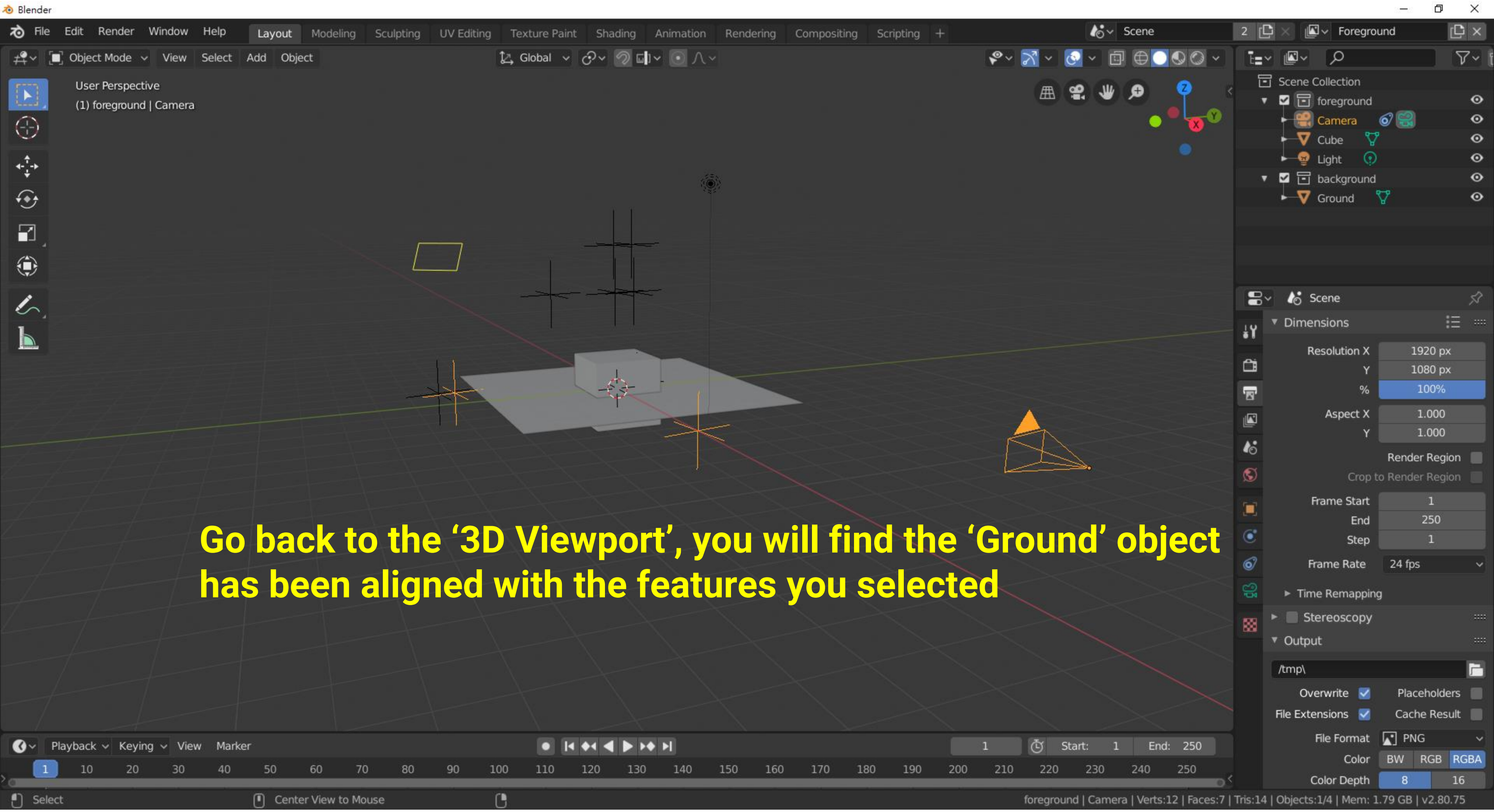


You can switch between the '3D Viewport' and 'Movie Clip Editor' for checking the positions of features

Setup Tracking Scene (6)

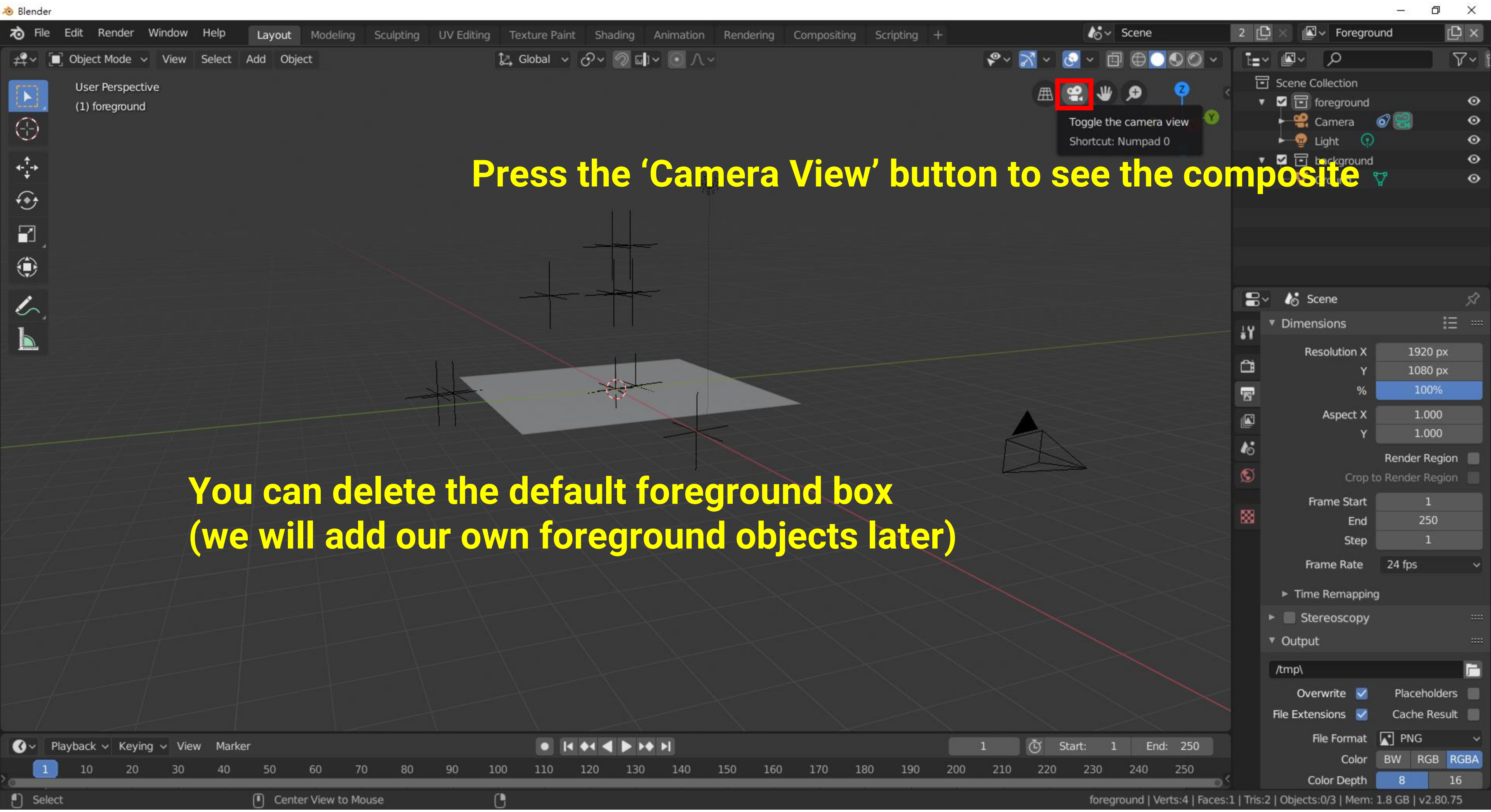


Setup Tracking Scene (7)

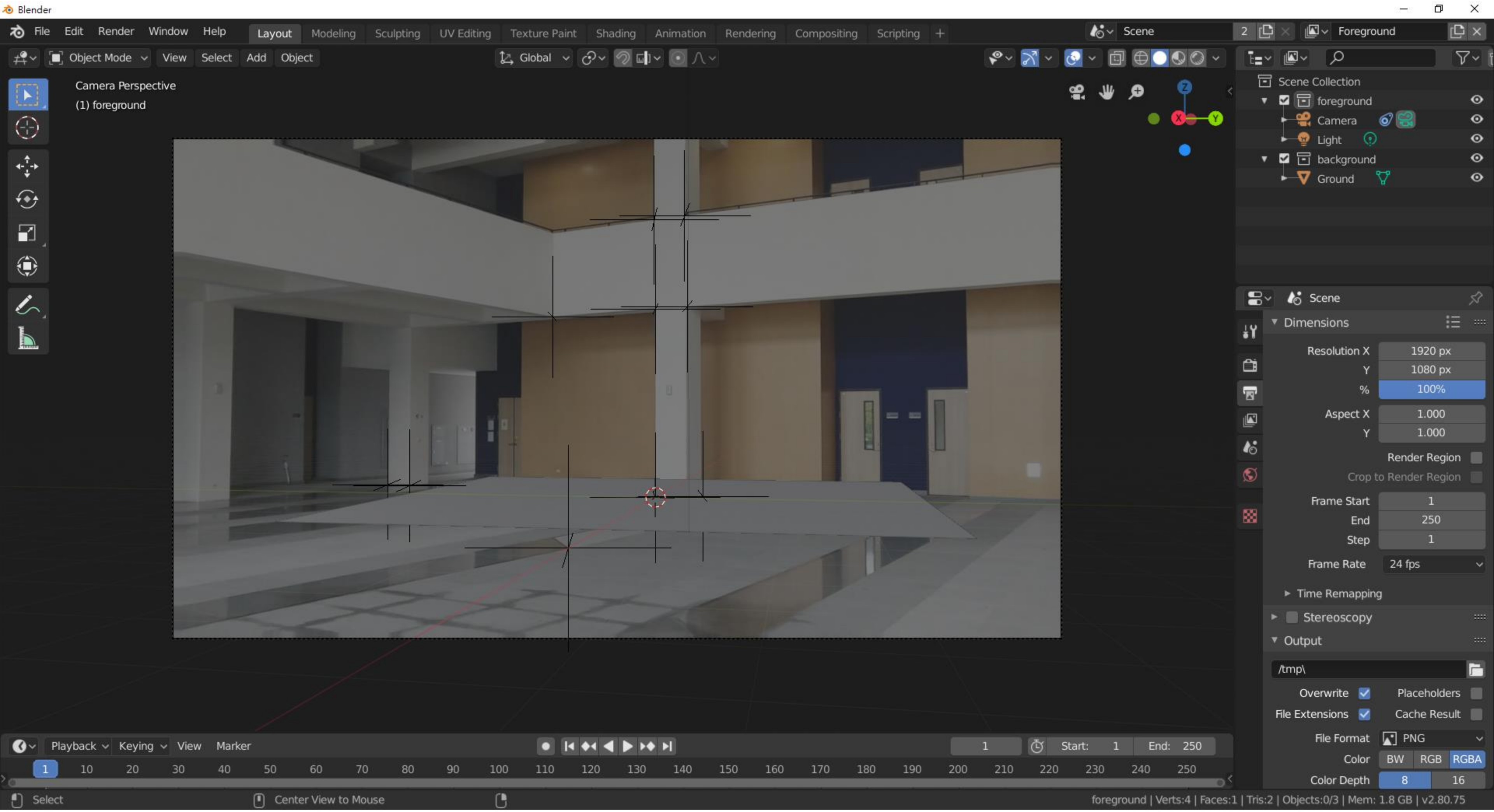


Go back to the '3D Viewport', you will find the 'Ground' object has been aligned with the features you selected

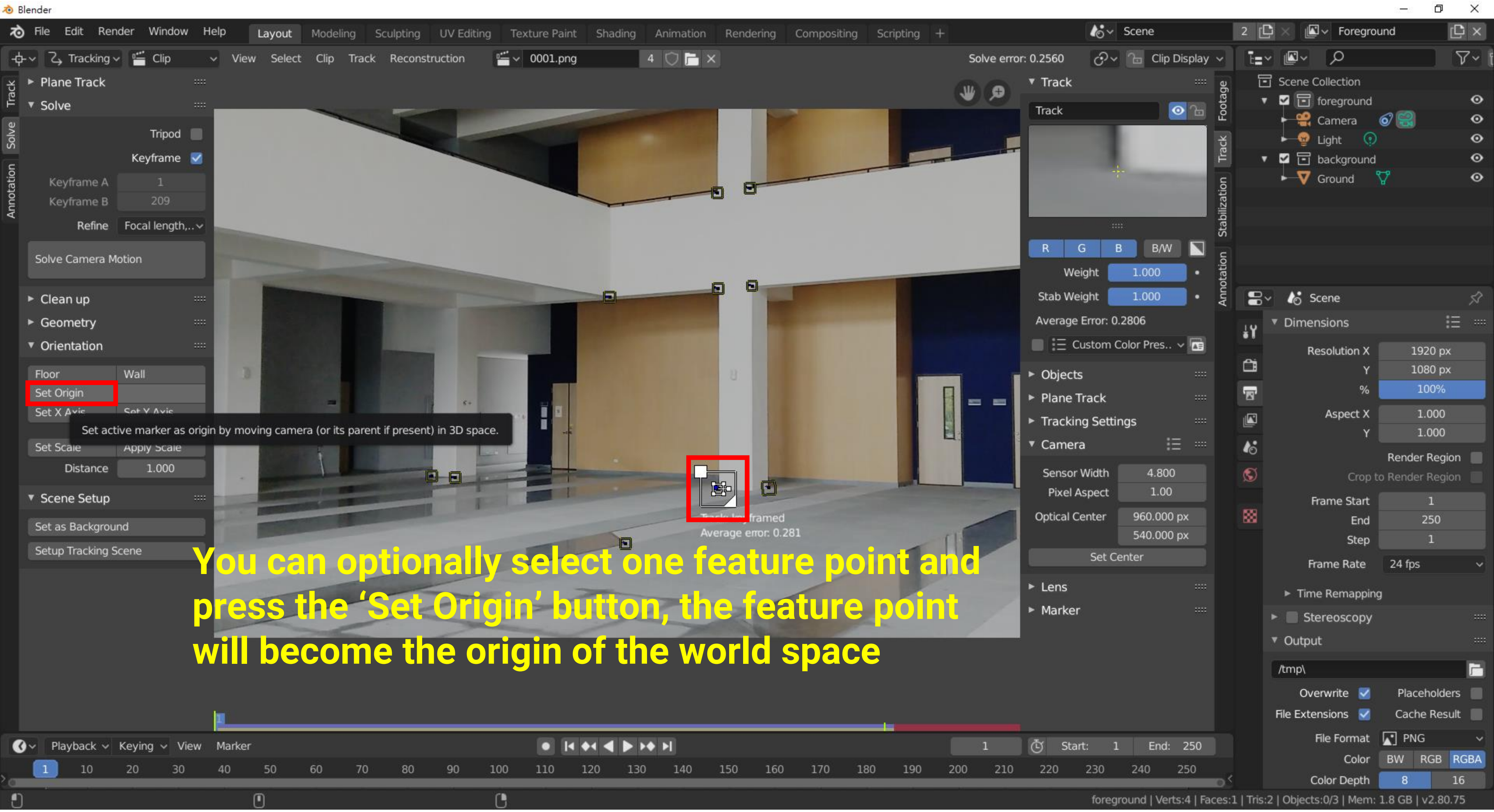
Setup Tracking Scene (8)



Setup Tracking Scene (9)

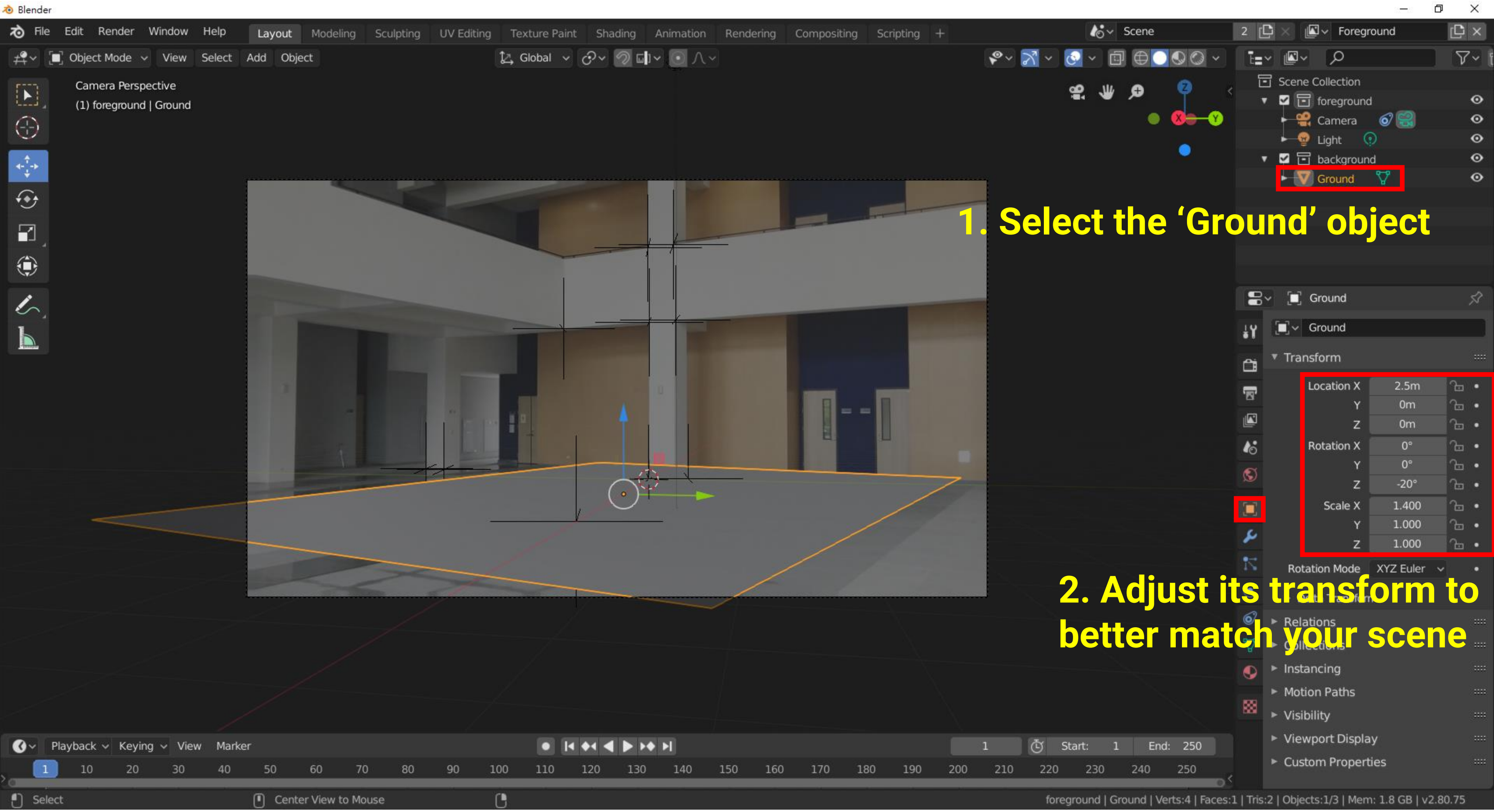


Setup Tracking Scene (10)

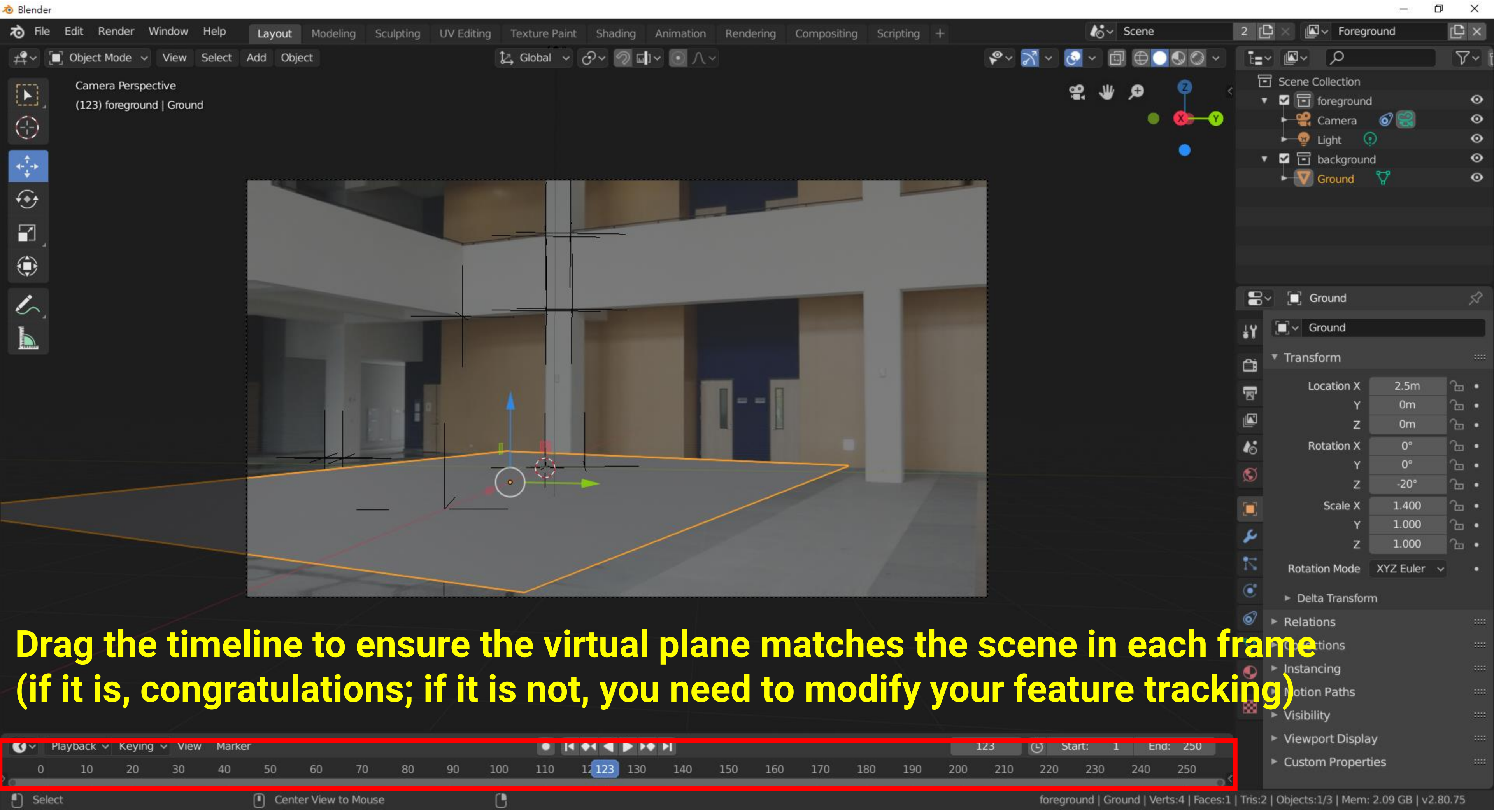


You can optionally select one feature point and press the 'Set Origin' button, the feature point will become the origin of the world space

Setup Tracking Scene (11)



Setup Tracking Scene (12)

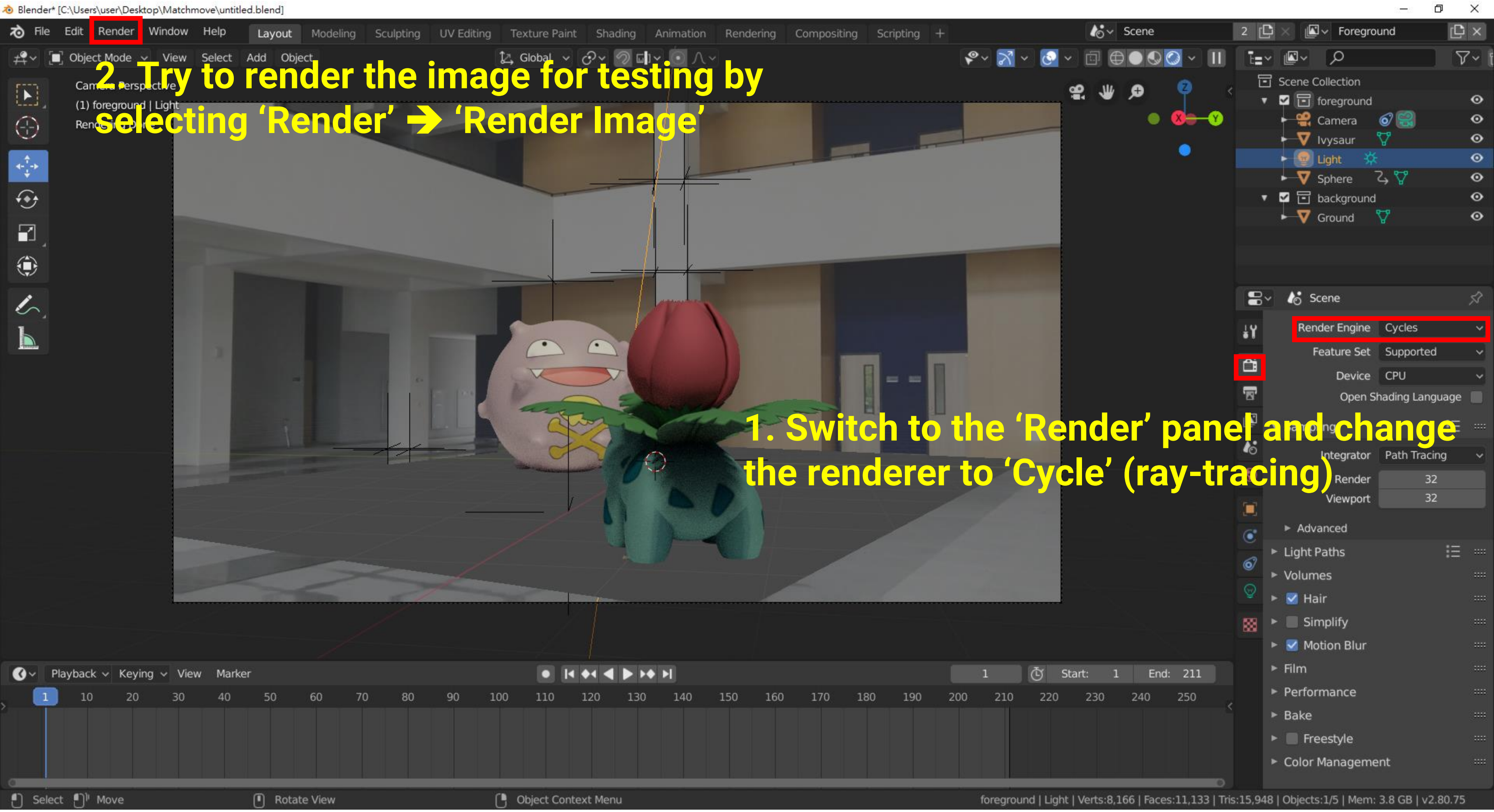


Drag the timeline to ensure the virtual plane matches the scene in each frame (if it is, congratulations; if it is not, you need to modify your feature tracking)

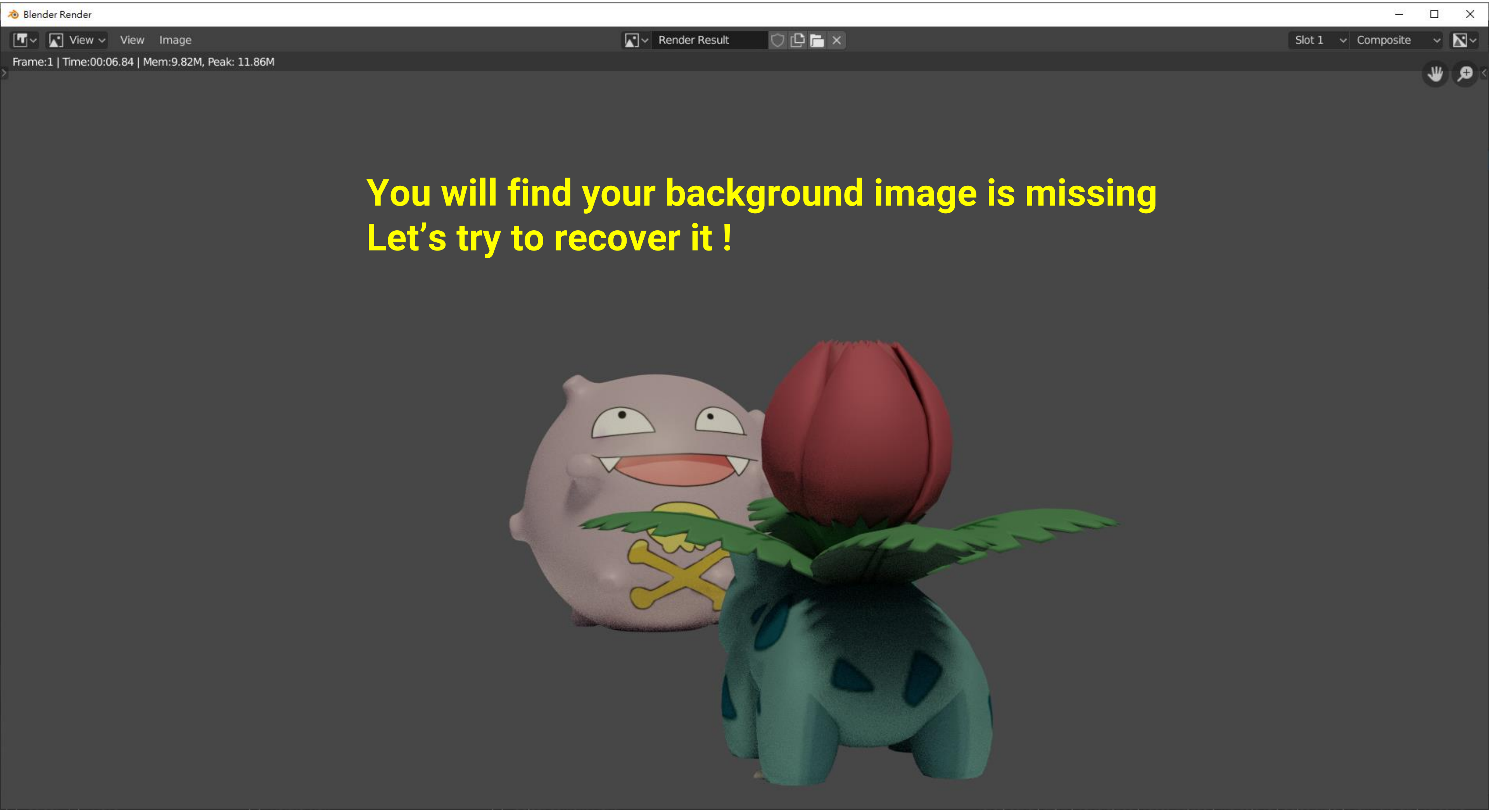
Add Virtual 3D Models (and Animations)

Add virtual 3D models and their animations

Try to Render the Frame (1)

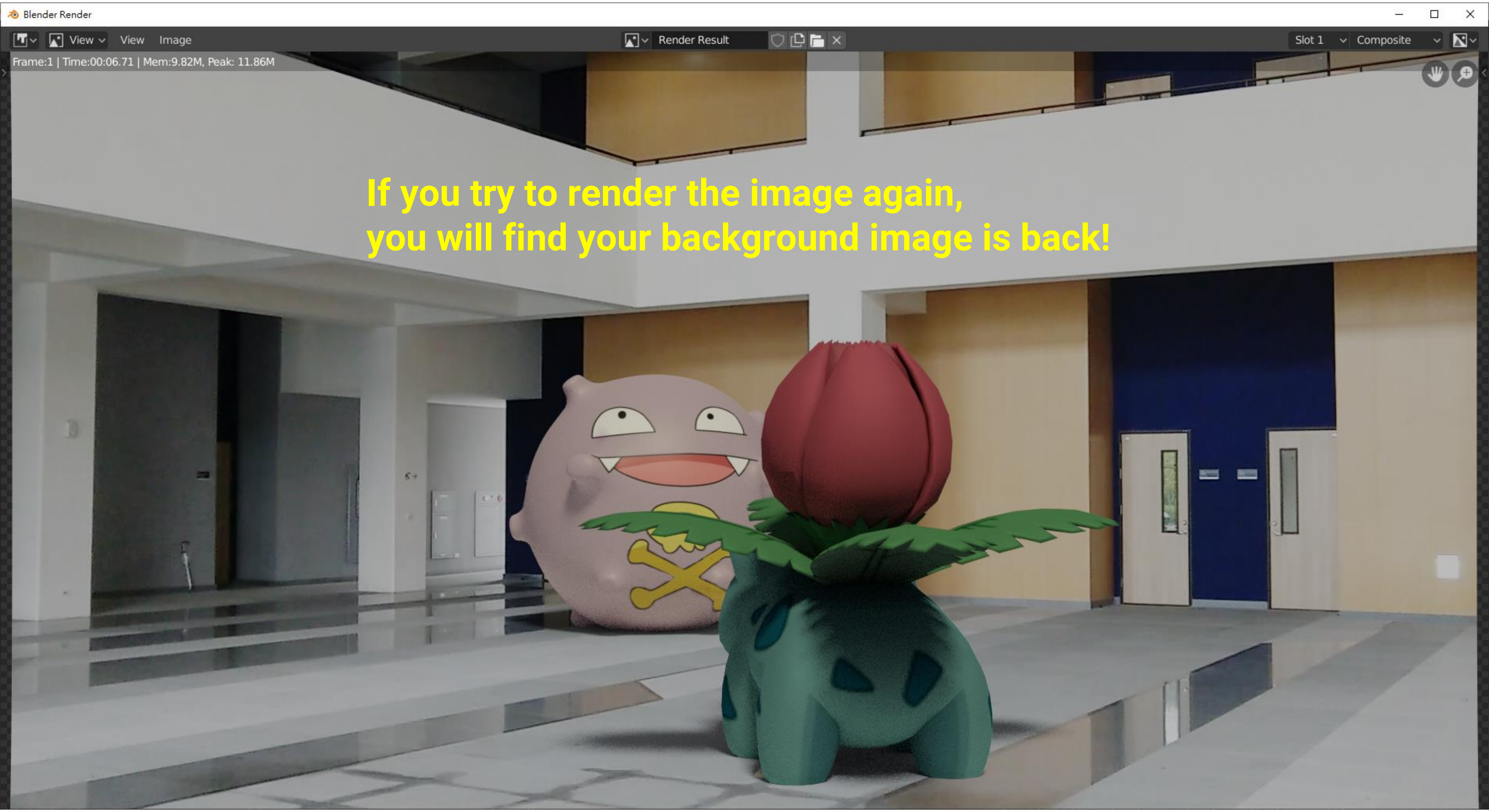


Try to Render the Frame (2)



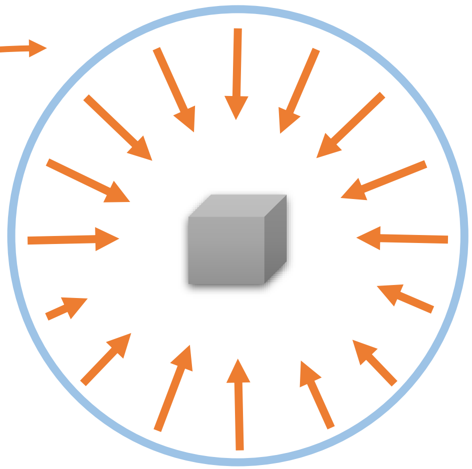
Try to Render the Frame (3)

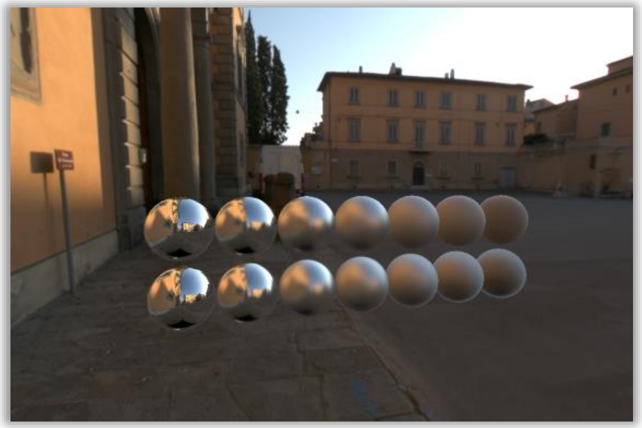
Try to Render the Frame (4)



Recap: Environment Lighting

- Environment light illuminates the scene from a **virtual sphere at infinite distance**
- The spherical energy distribution is usually represented with longitude-latitude images
- Also called **image-based lighting (IBL)**



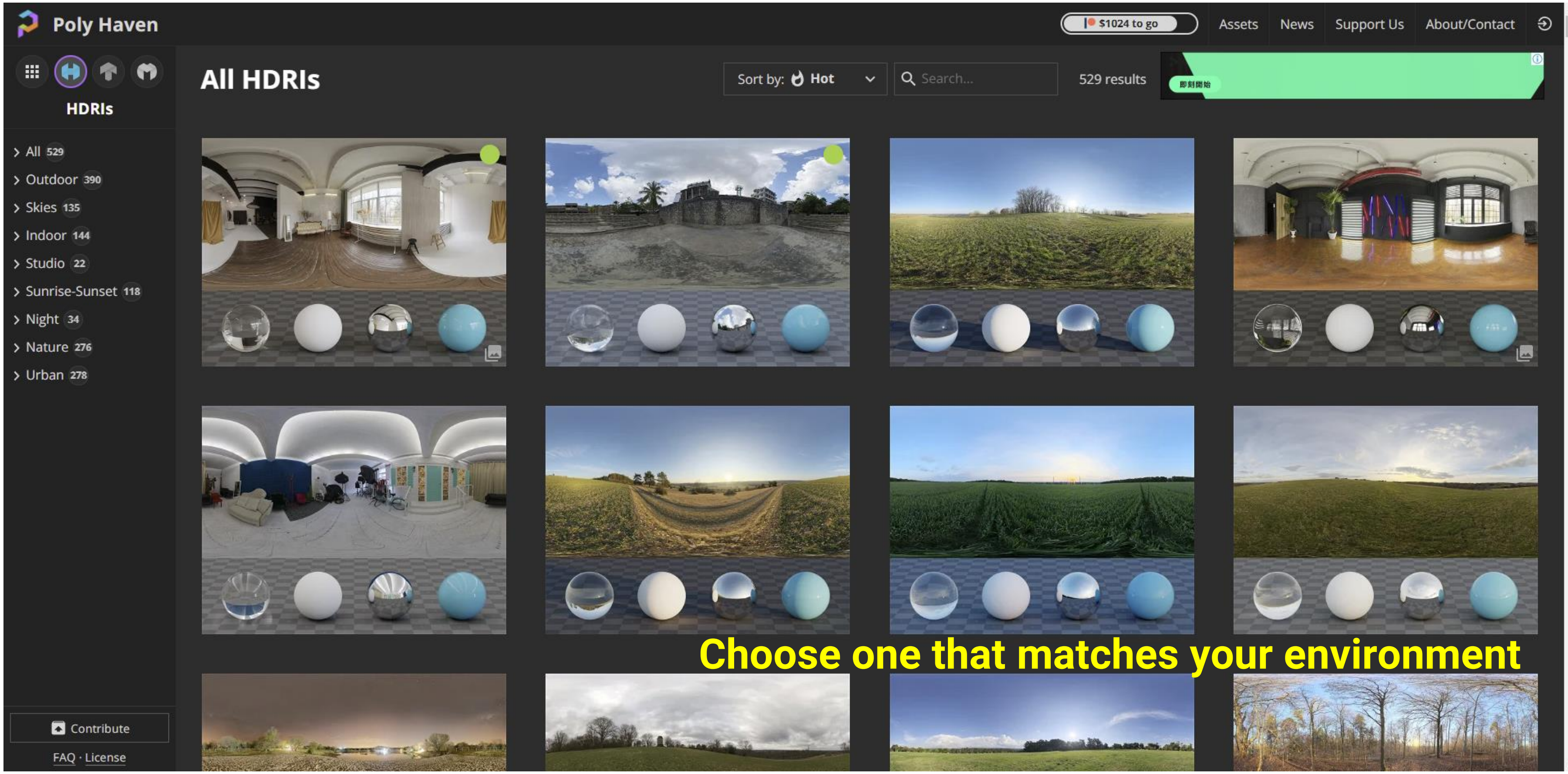


Recap: Environment Lighting

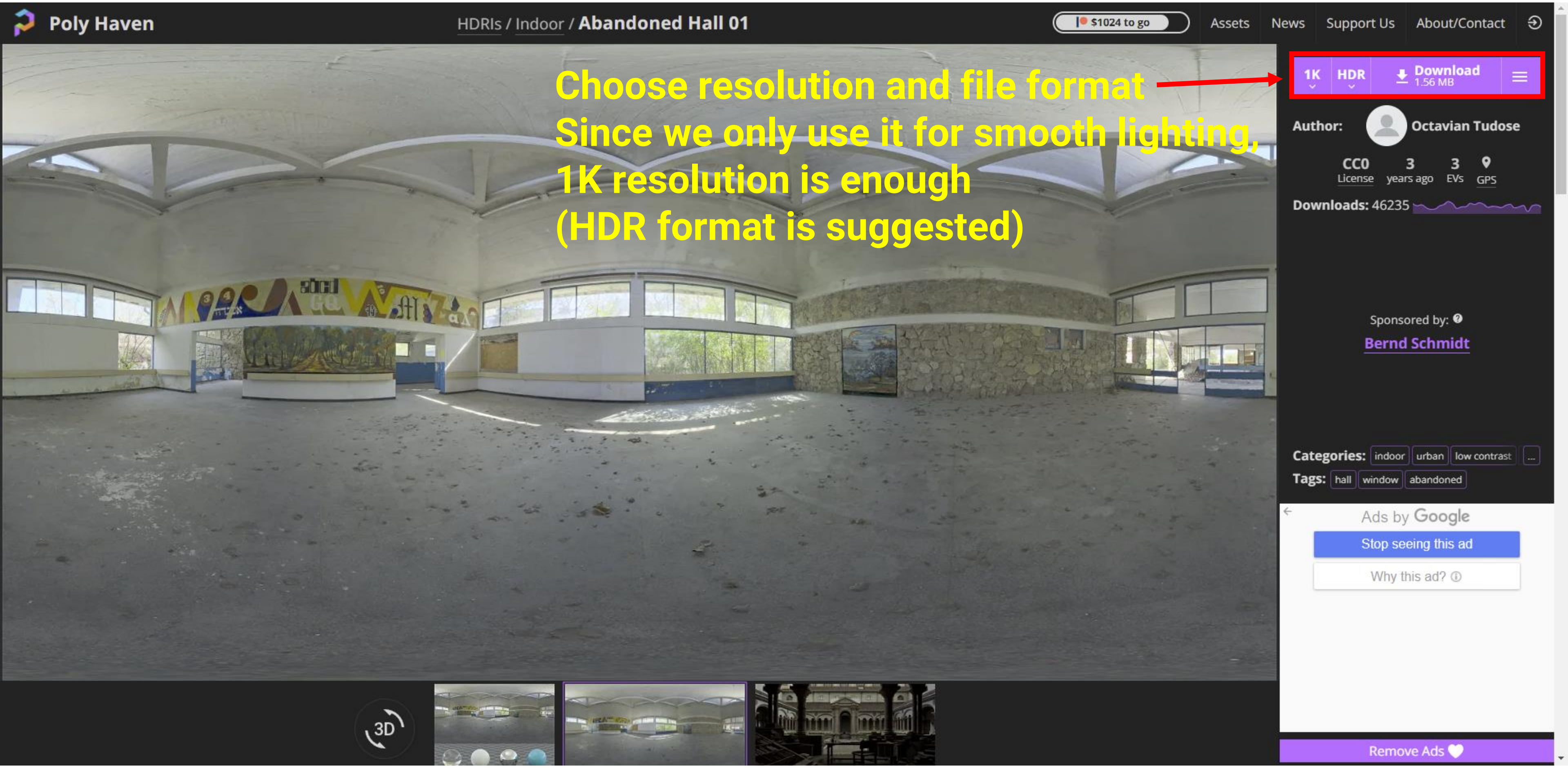
- Widely used in digital visual effects and film production

Environment Lighting Resource

- Download free HDR environment map on the internet
  https://polyhaven.com/hdris

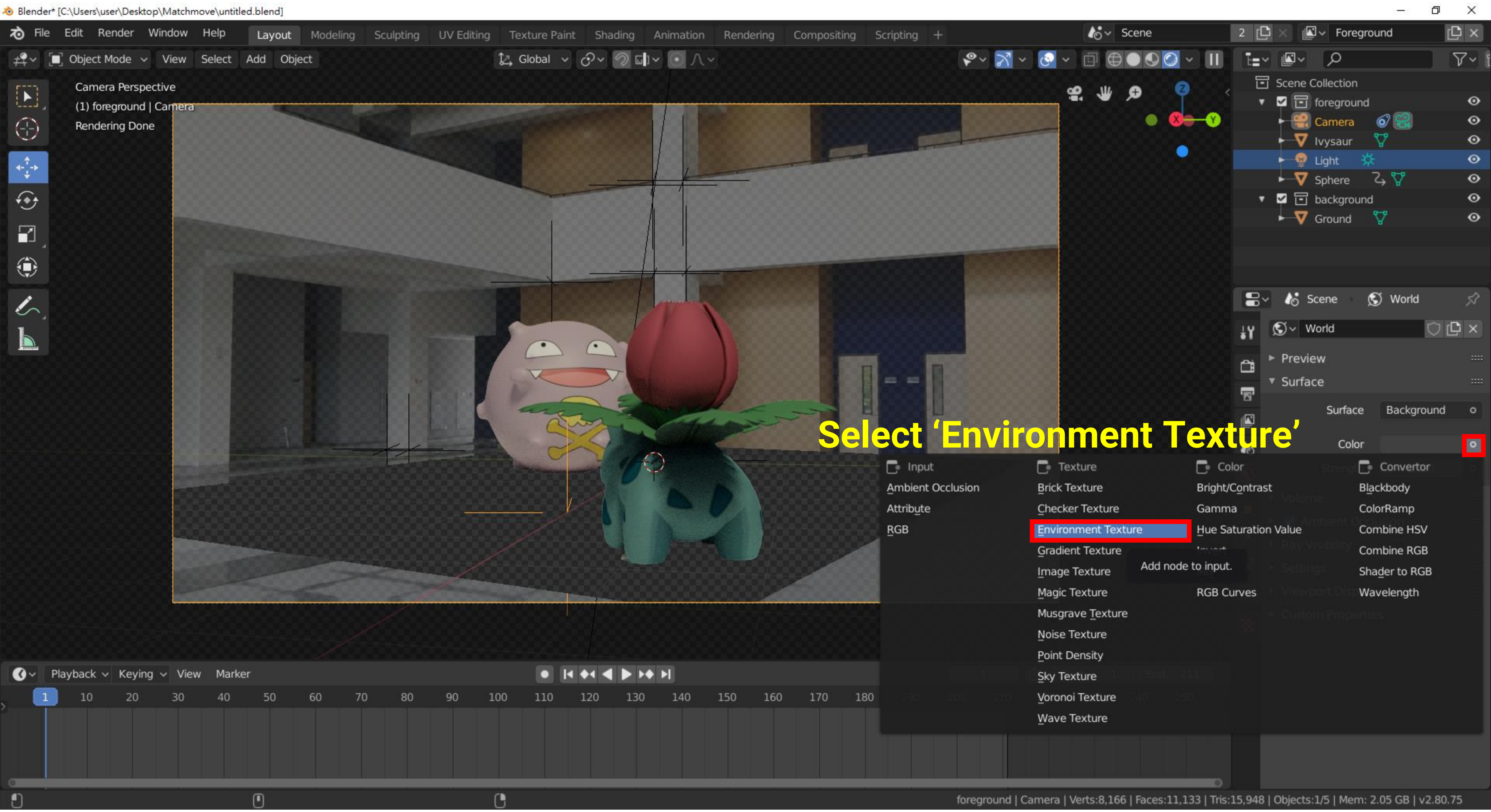


Environment Lighting Resource (cont.)

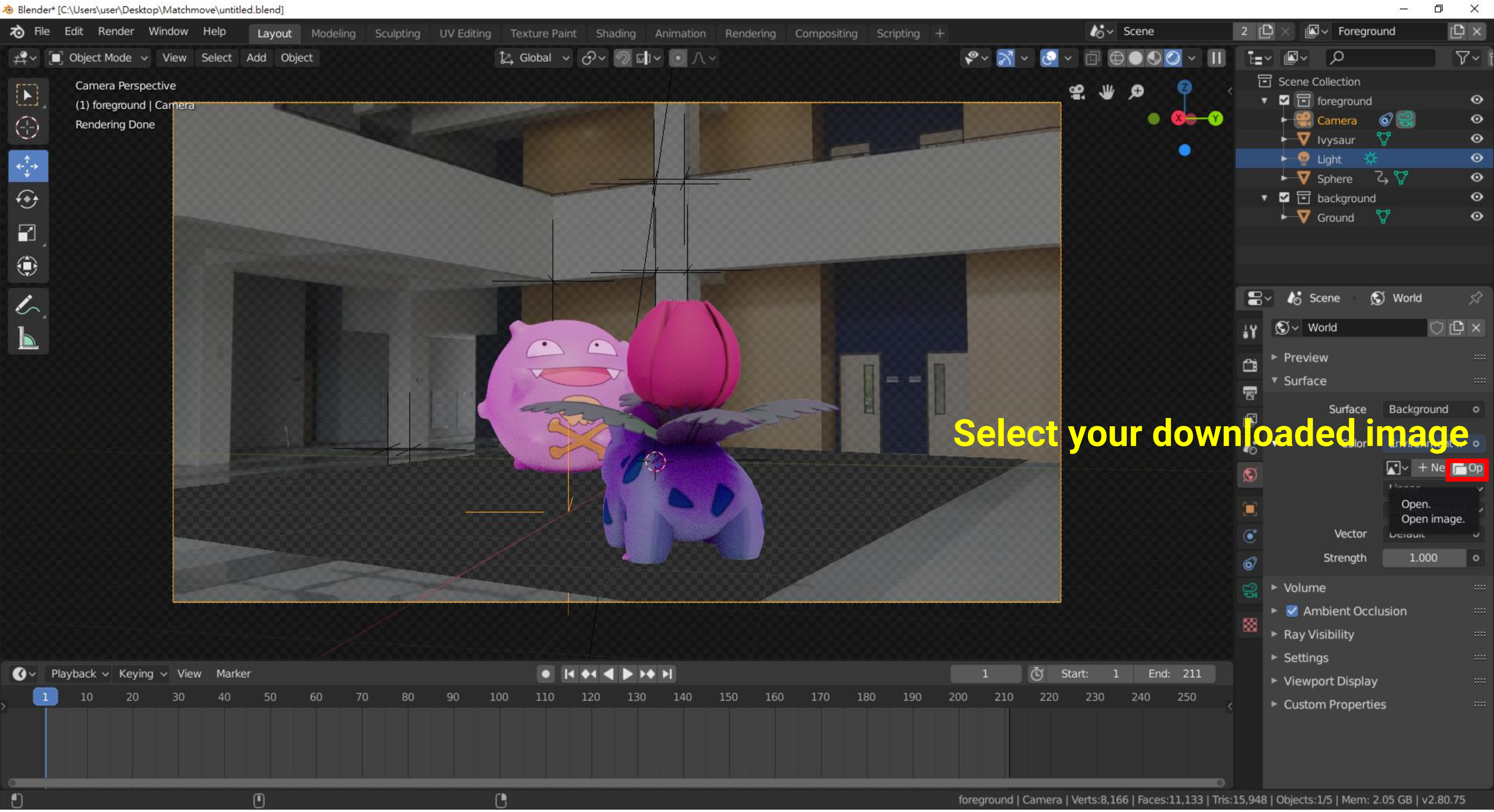


Add more realistic lighting (1)

Add more realistic lighting (2)



Add more realistic lighting (3)



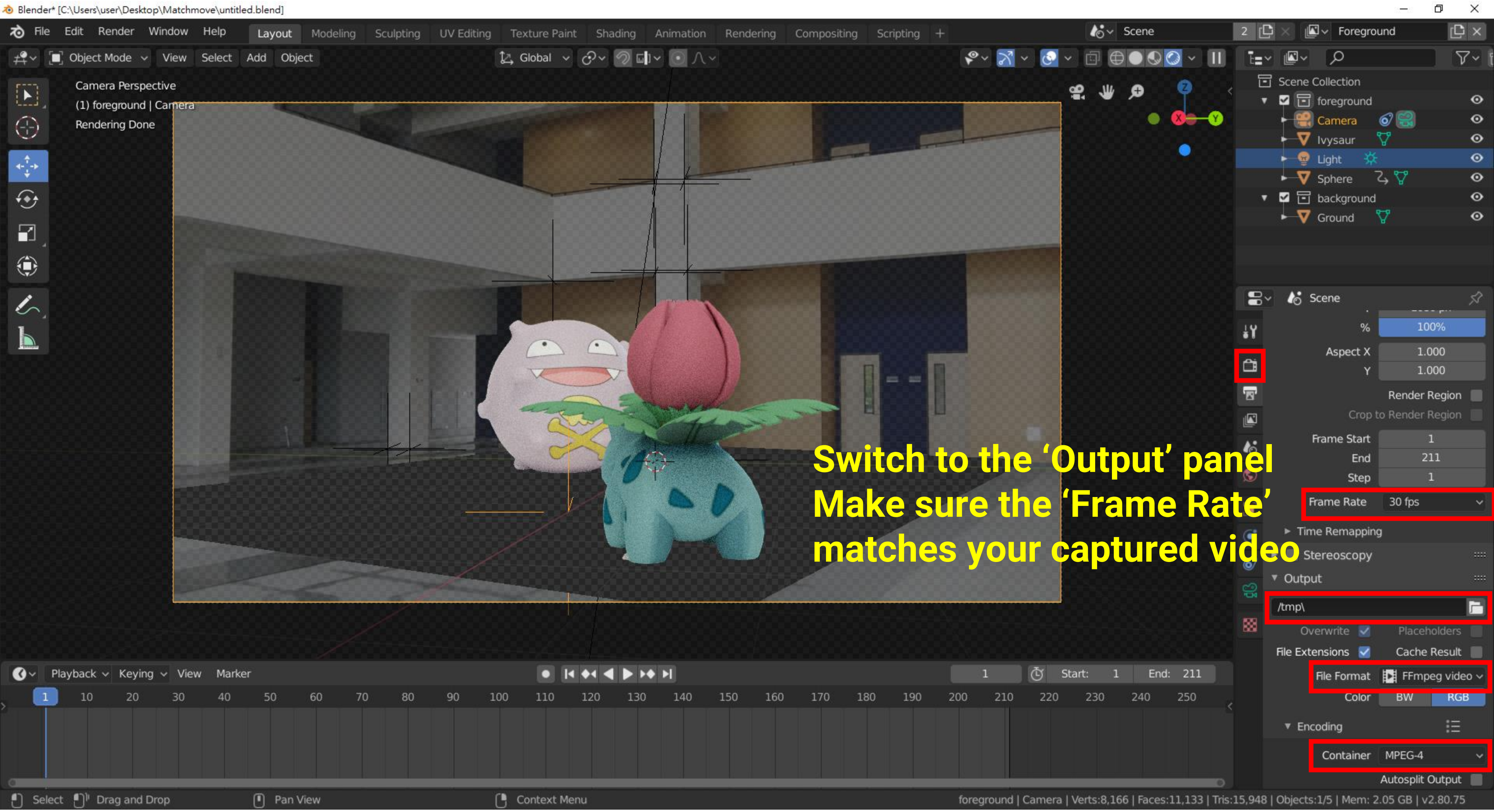
Add more realistic lighting (4)

The models are now shaded by the original light plus an environment map

You can adjust the strength of the environment lighting

Output Composite Video

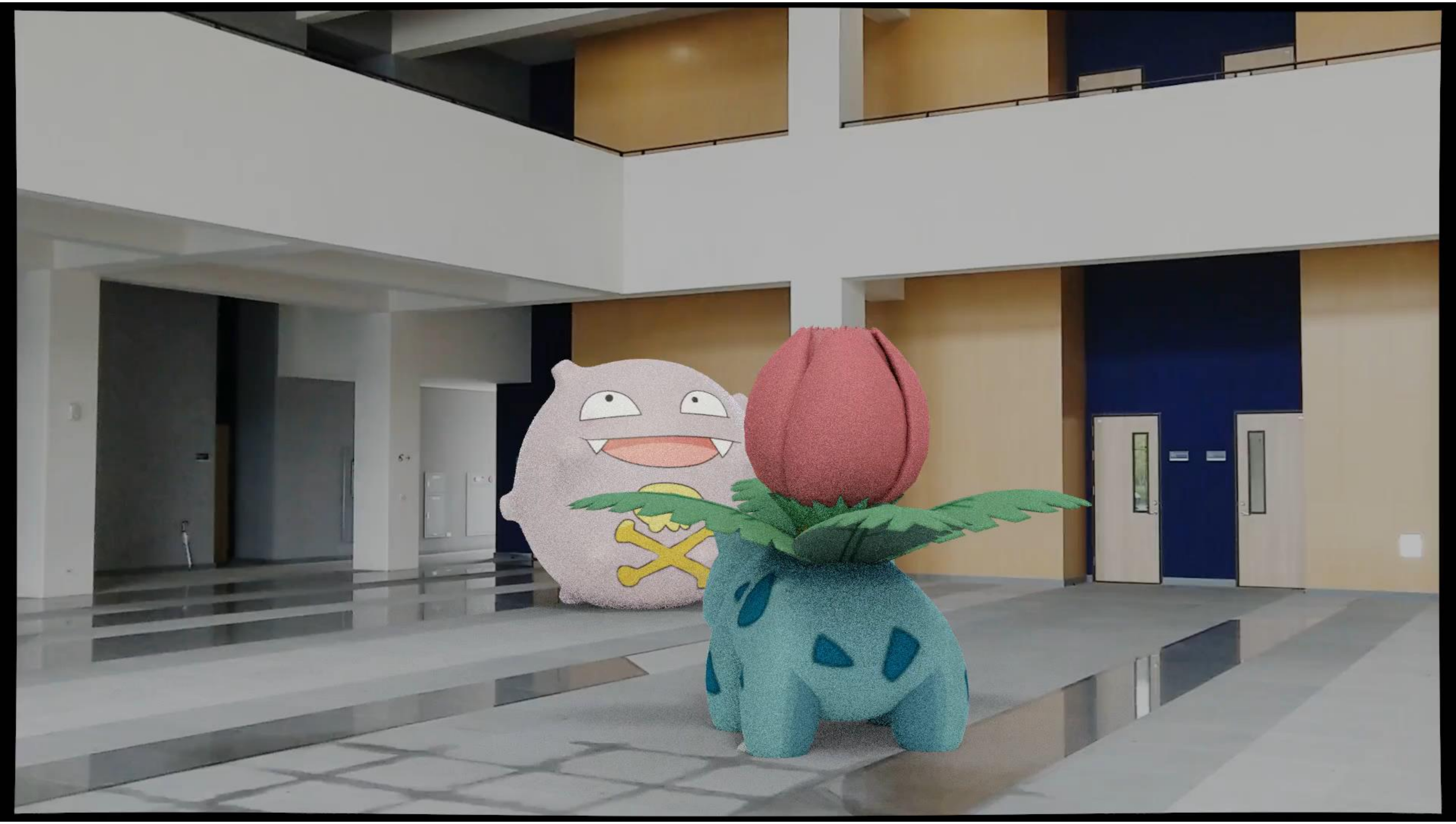
Set output configuration (1)



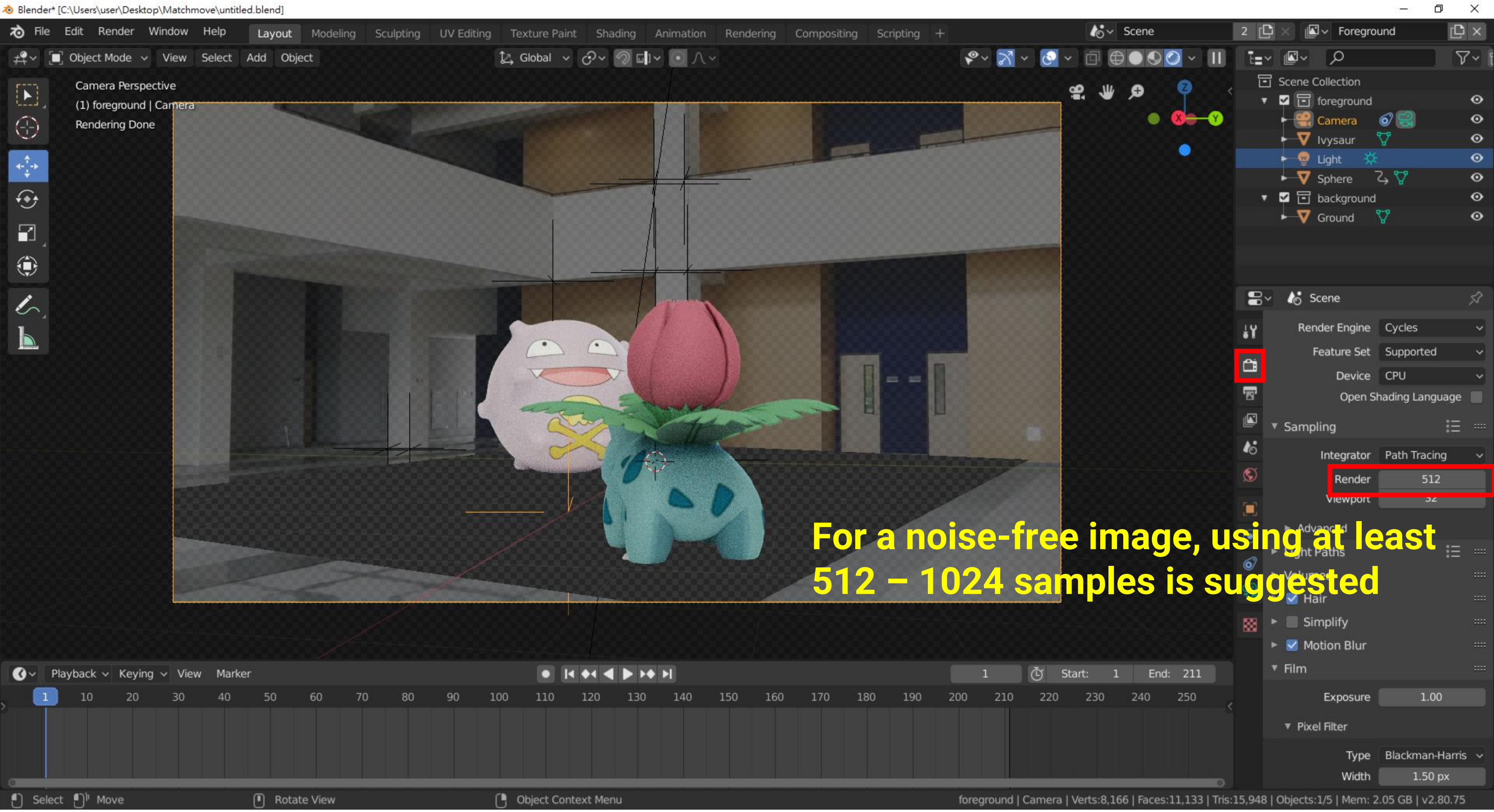
Set output configuration (2)

Render animation preview

Preview (check the poses and animations of the models)



Set to high-quality rendering



Final Output