Northern Illinois University

Extrinsic Calibration of Two Overhead Cameras

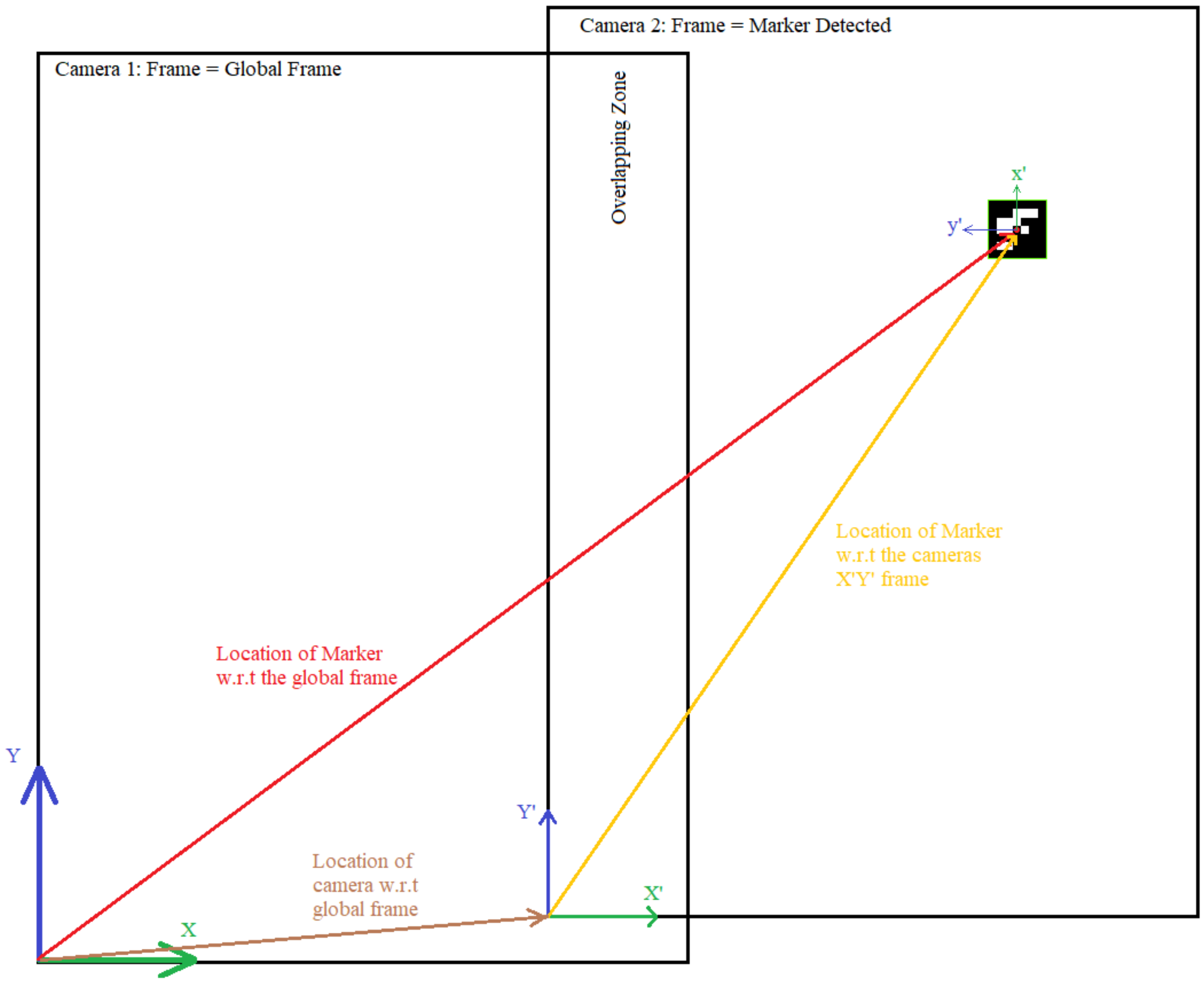
References CVTracking\Tracker.py function: getCameraRelations(camObj1, camObj2)

Contributor: zTaylor-Robotics

Purpose

- Determine the location and rotation of cameras with respect to one another.
- Project implementation:
  - Aruco marker is detected in one camera and is <u>related</u> back to the global frame defined by a cameras frame.
  - This <u>relation</u> is the location and rotations of multiple cameras that when combined, result in a transformation from the camera to the global frame.
  - Then the location and orientation of the marker is combined with the transformation to get its location and orientation w.r.t the global frame.
  - The next slide shows a markers relation to a camera frame (Yellow vector) being transformed into a relation with the global frame (Red vector). The transformation being the combination of the brown and yellow vector.

Purpose Continued

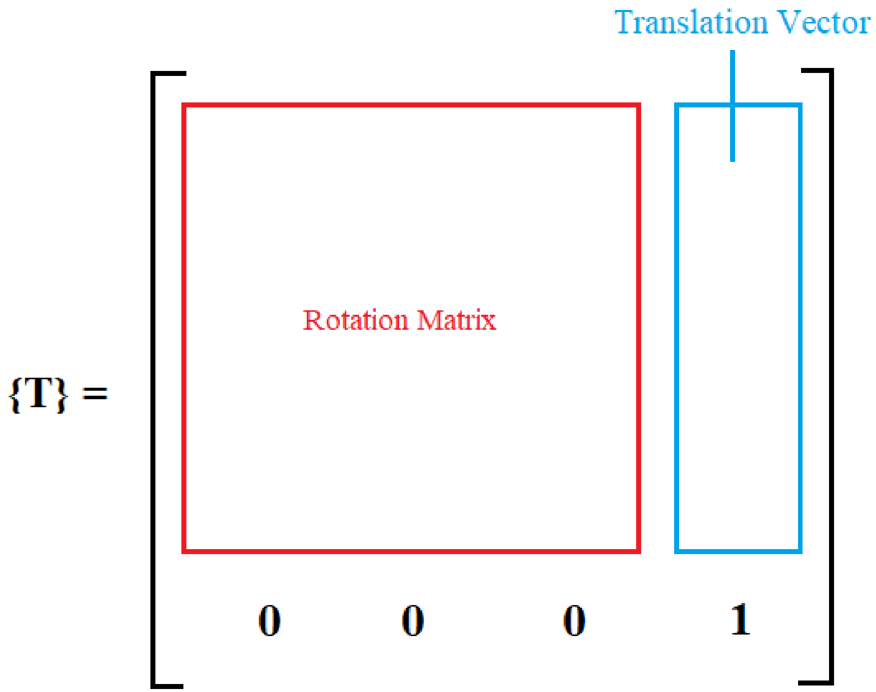


Mathematics Behind the Purpose

- The mathematics begins with the introduction of the rotation and translation matrix {T} shown below:
  - The rotation matrix is a 3x3 matrix which when applies 3x1 vector, rotates that vector by a specified theta.
  - The Translation vector is a 3x1 vector whose values represent a x, y, z translation.
- This matrix represents a rotation and translation of an object in space w.r.t a given reference frame.

<u>Matrix Notation:</u>

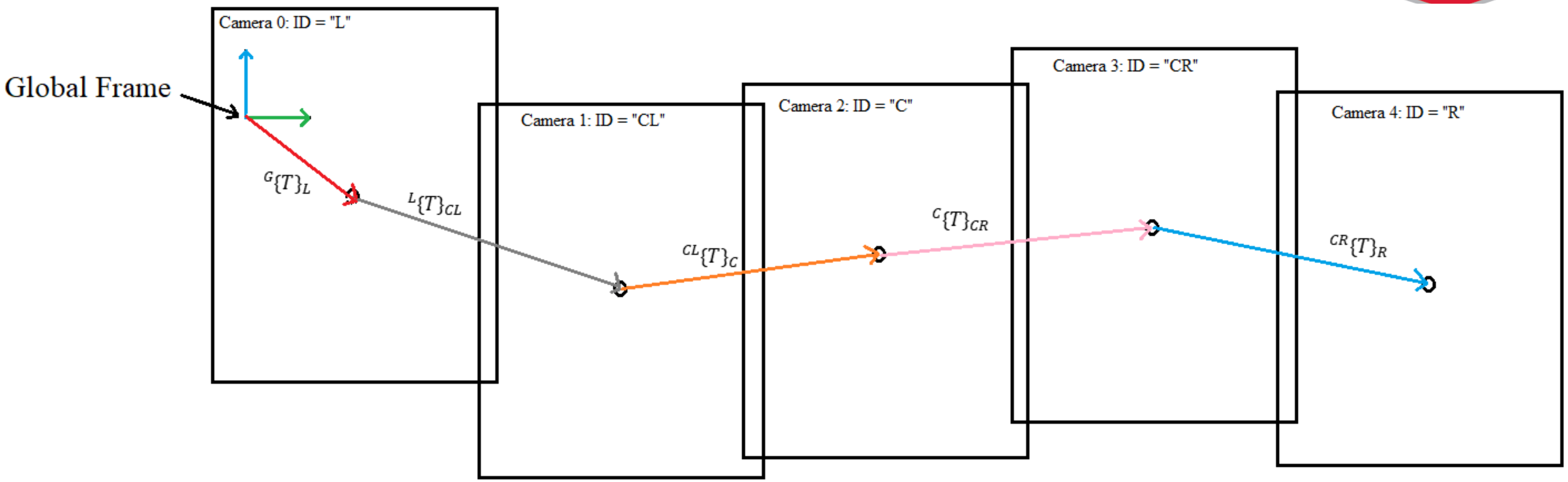
- ${}^{CF}\{T\}_{aruco}$
  - CF = Camera Frame
  - aruco = Fiducial marker
  - This notation represents the aruco marker's location and orientation w.r.t the Camera frame
- ${}^{G}\{T\}_{CF}$
  - G = global frame
  - This notation represents the camera frame's location and orientation w.r.t the global frame
- ${}^{G}\{T\}_{aruco} = {}^{G}\{T\}_{CF} \ * \ {}^{CF}\{T\}_{aruco}$
  - The transformation of the aruco w.r.t the global frame is found by pre-multiplying the aruco w.r.t the camera frame with the camera frame w.r.t the global frame.



Calibration Purpose

- The extrinsic calibration seeks to determine the location and orientation of the camera frame w.r.t the global frame.
  - Represented by ${}^{G}\{T\}_{CF}$
- The approach to finding this value varies based on the setup of the system.
  - The system in the project is represented in the following power point slide.
  - The project consists of 5 cameras that are staggered within a room to track moving objects in a large global frame.
  - The cameras are such that to get ${}^{G}\{T\}_{CF}$, the transformation from one camera to another is needed.
  - The equation representing this transformation is shown in the following slide as well.

Deriving ${}^{G}\{T\}_{CF}$ for Each Camera

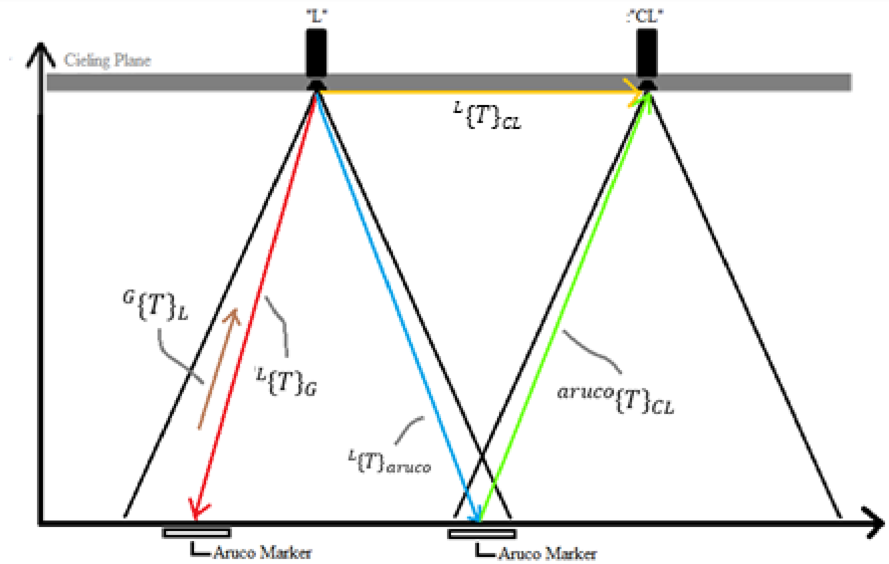


- ${}^{G}\{T\}_{L}$ = "L" cameras transformation to the corner of the frame
  - Each rectangle is a camera plane that rotated 90^{0} necessary to all be orientated similar to each other (rotation of each camera is slightly different)
  - ${}^{G}\{T\}_{L}$ is found by placing an aruco marker and determining the cameras pose and orientation to that marker.
  - ${}^{G}\{T\}_{L}$ can be determined later as seen in the next slide.

Deriving ${}^{G}\{T\}_{CF}$ for Each Camera

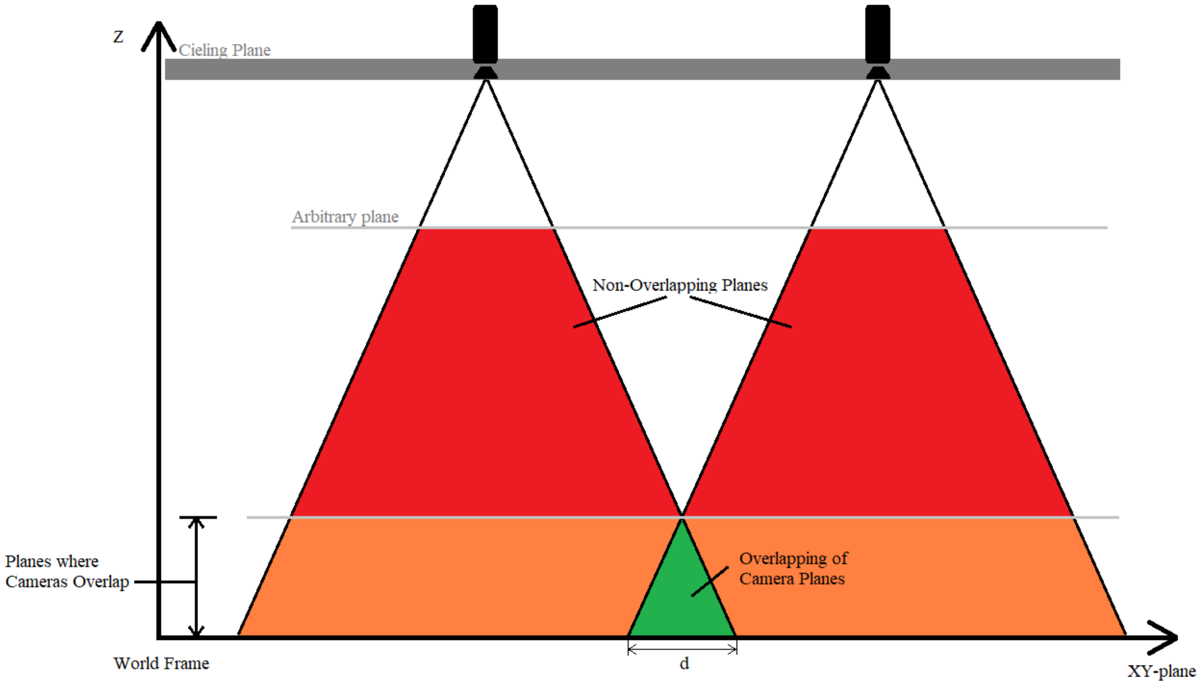
- ${}^{G}\{T\}_{CL} = {}^{G}\{T\}_{L} * {}^{L}\{T\}_{CL}$
- ${}^{G}\{T\}_{C} = {}^{G}\{T\}_{L} * {}^{L}\{T\}_{CL} * {}^{CL}\{T\}_{C}$
- ${}^{G}\{T\}_{CR} = {}^{G}\{T\}_{L} * {}^{L}\{T\}_{CL} * {}^{CL}\{T\}_{C} * {}^{C}\{T\}_{CR}$
- ${}^{G}\{T\}_{R} = {}^{G}\{T\}_{L} * {}^{L}\{T\}_{CL} * {}^{CL}\{T\}_{C} * {}^{C}\{T\}_{CR} * {}^{CR}\{T\}_{R}$
- The equations above are a part of the extrinsic calibration process for the system shown above.
- The all camera objects will only contain one transformation matrix, which relates that camera to the camera next to it.
  - Example: Camera "CL" will have the ${}^{L}\{T\}_{CL}$ matrix
- On tracking object startup, the cameras being used will be specified in Left to Right order for which ${}^{G}\{T\}_{CF}$ will for each camera will be stored in a list that will be a property of the tracking object.
  - Storage convention => ${}^{C}\{T\}_{CR}$ = CR_wrt_C.txt

Determining Camera Pose with an Aruco Marker



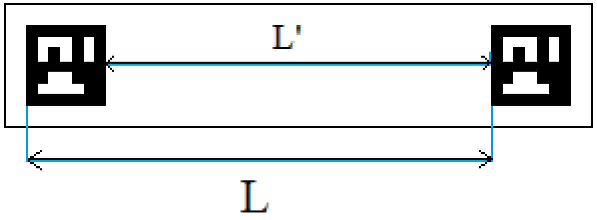
- The figure above represents the information we receive from the arucoDetect() function.
- We get the aruco markers position and orientation w.r.t the camera
- For calibration we will use the inverse of $^{CF}\{T\}_{aruco}$ to achieve:
- $(^{CF}\{T\}^{aruco})^{-1} = {}^{aruco}\{T\}_{CF}$
  - As shown above, aruco = global frame G
- This then allows us to get the relationship between cameras by placing the aruco marker between each camera as shown above.
- We can then use the formula: $^{L}\{T\}_{aruco} * {}^{aruco}\{T\}_{CL} = {}^{L}\{T\}_{CL}$
- The whole process is then mapped as: $^{G}\{T\}_{CL} = inv(^{L}\{T\}_{G}) * {}^{L}\{T\}_{aruco} * inv(^{CL}\{T\}_{aruco})$ for the above example
- The camera be found using one of the two methods mentioned in the next few slides.

First Method: Single Aruco to get Camera Relations



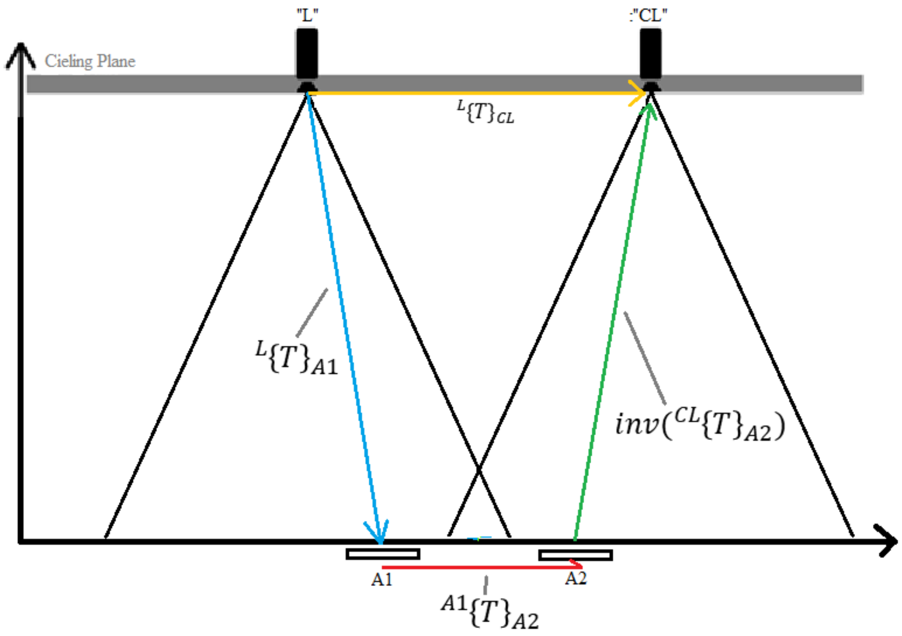
- The following process assumes the following situation:
  - Two overhead cameras looking down which have two viewing planes which overlap at some point which results in a 2D plane with cross distance d.
    - If the above condition isn't met or if the overlap is too small for a fiducial marker, an alternative approach will be shown later using the file CVTracking\Samples\extrinCal.docs. In Setup: Second Method
  - Length d is such that an Aruco marker can fit and be picked up by both cameras.
- This directly relates to the calculations shown in the previous slide.

Second Method: Two Aruco Markers to get Camera Relations



- In the situation where distance d results in the inability to capture the aruco marker within two frames, two aruco markers will be needed.
- The set up of the aruco markers are shown above.
- They are set up so that the transformation from one marker to another is a perfect translation (single dimension) of length, L, from one to the other.
- This is such that the orientation of each marker is perfectly the same.

Second Method: Two Aruco Markers to get Camera Relations



- The mathematics behind this change is shown in the image above.
- $^{G}\{T\}_{CL} = {}^{G}\{T\}_{L} * {}^{L}\{T\}_{A1} * {}^{A1}\{T\}_{A2} * inv(^{CL}\{T\}_{A2})$