COGNEX

AlignPlus® 4.3

Concepts

2020 December 16

Legal Notices

The software described in this document is furnished under license, and may be used or copied only in accordance with the terms of such license and with the inclusion of the copyright notice shown on this page. Neither the software, this document, nor any copies thereof may be provided to, or otherwise made available to, anyone other than the licensee. Title to, and ownership of, this software remains with Cognex Corporation or its licensor. Cognex Corporation assumes no responsibility for the use or reliability of its software on equipment that is not supplied by Cognex Corporation. Cognex Corporation makes no warranties, either express or implied, regarding the described software, its merchantability, non-infringement or its fitness for any particular purpose.

The information in this document is subject to change without notice and should not be construed as a commitment by Cognex Corporation. Cognex Corporation is not responsible for any errors that may be present in either this document or the associated software.

Companies, names, and data used in examples herein are fictitious unless otherwise noted. No part of this document may be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, nor transferred to any other media or language without the written permission of Cognex Corporation.

2020 December 16 9:19 AM

Portions of the hardware and software provided by Cognex may be covered by one or more U.S. and foreign patents, as well as pending U.S. and foreign patents listed on the Cognex web site at: cognex.com/patents.

The following are registered trademarks of Cognex Corporation:

Cognex, 2DMAX, Advantage, AlignPlus, Assemblyplus, Check it with Checker, Checker, Cognex Vision for Industry, Cognex VSOC, CVL, DataMan, DisplayInspect, DVT, EasyBuilder, Hotbars, IDMax, In-Sight, Laser Killer, MVS-8000, OmniView, PatFind, PatFlex, PatInspect, PatMax, PatQuick, SensorView, SmartView, SmartAdvisor, SmartLearn, UltraLight, Vision Solutions, VisionPro, VisionView

The following are trademarks of Cognex Corporation:

The Cognex logo, 1DMax, 3D-Locate, 3DMax, BGAII, CheckPoint, Cognex VSoC, CVC-1000, FFD, iLearn, In-Sight (design insignia with cross-hairs), In-Sight 2000, InspectEdge, Inspection Designer, MVS, NotchMax, OCRMax, PatMax RedLine, ProofRead, SmartSync, ProfilePlus, SmartDisplay, SmartSystem, SMD4, VisiFlex, Xpand

Table of Contents

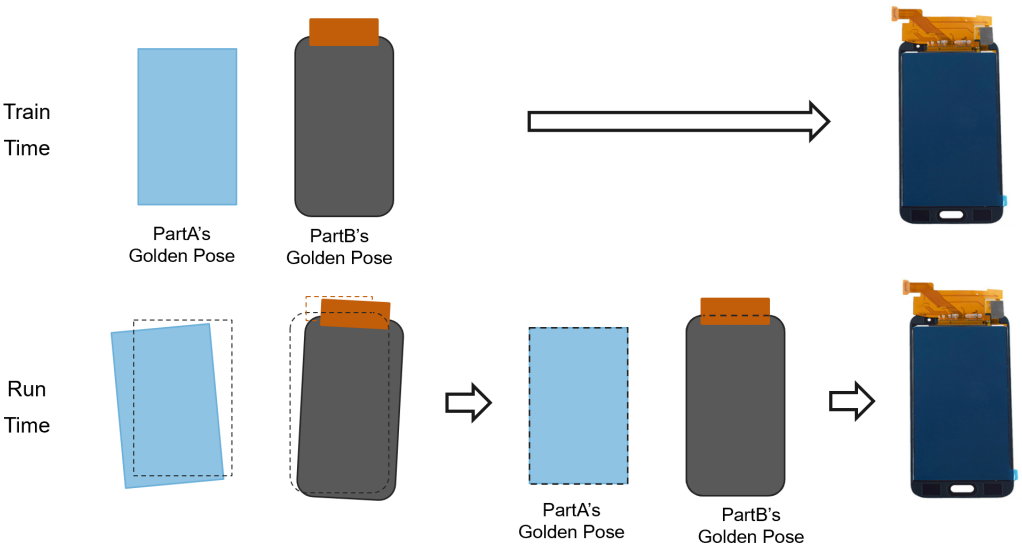
Application Type

Alignment

Alignment Concept

Alignment refers to applications wherein a motion device such as a XYΘ stage or a gantry system is moved based upon the current position of work object(part) and comparing it to the part's trained position (golden pose) to align the object to desired pose.

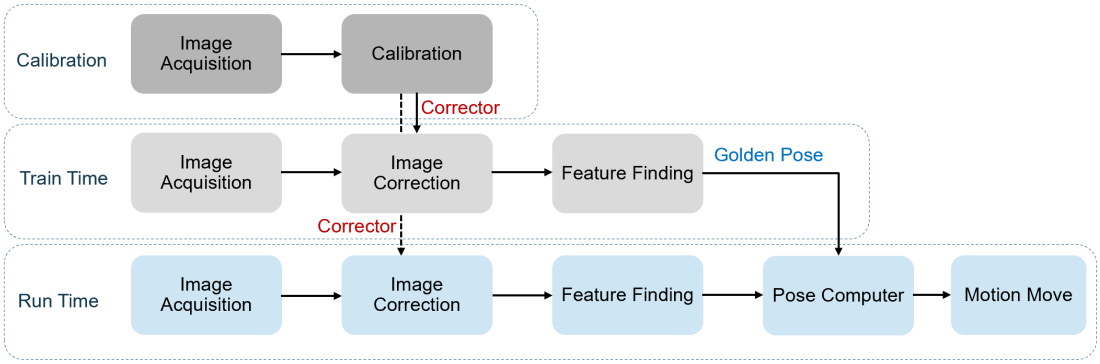
In the example blow, Part A and Part B's positions are trained at their own stations during train time. In run time, both Part A and Part B align to their trained golden poses respectively and subsequently attached.



The vision system's task is to locate one or more features of the part that are used to calculate the part's pose (Translation and Rotation). Those features could be corner points, multiple edges, or registration marks/fiducials. By comparing the difference between run time and train time poses, the vision system calculates and communicates the necessary movements to the motion devices.

Alignment Procedure

Accurate alignment results from successful execution of three key steps: (1) Calibration, (2) Train Time, and (3) Run Time. Calibration and Train Time tasks are only executed during setup, while the run time tasks are executed during production operation. The purpose of calibration is to establish shared coordinates among different cameras, and motion devices. Calibration only needs to be performed once if the machine physical setup is held constant (as long as there is no hardware change such as camera position change, lens change, or station's height change which leads to calibration data becoming invalidated). Train time tasks are executed to teach the target position(golden pose) of part relative to the motion device, it's also only required once as long as there is no change on calibration data or trained golden pose. Run time tasks are executed every time when vision system needs to guide motion for completing the alignment task.



Calibration

The shared coordinates among different cameras, and motion devices are established mainly through hand-eye calibration.

Before hand-eye calibration, a calibration target such as checkerboard plate or run time part with fiducial marks is placed on platform. During calibration, either the movable platform moves and rotates the calibration target or cameras move and rotate to a set of predefined poses. At each step of this process, the vision system acquires images and accumulate features of the calibration target. At the end of the calibration, the vision system runs hand-eye calibrator to calculate the transforms among different cameras and motion devices, and establishes a shared coordinate space called Home2D.

This transform between Home2D and raw image coordinate spaces for each camera is then stored within a structure called Corrector though which any raw image in train time or run time captured by the same camera can be converted into corrected image with Home2D coordinate space attached. With the corrected images, the vision system can find features and output their locations in Home2D space directly which then will be used for computing how much motion device should move to complete alignment.

Train Time

The purpose of train time is to set up target pose(golden pose) for run time part. Golden pose is consisted of a list of features and their coordinates in Home2D. The steps are:

1. Manually move a part to the ideal pose with respect to motion device
2. Acquire image(s) of the part
3. Use Correctors generated in calibration process to convert raw images into corrected images
4. Configure vision task to find the features on that part
5. Send command to train golden pose. This golden pose will be saved within the vision system.

Run Time

Note: When it comes to run time, Correctors and golden pose should already be available and saved in recipe. If not, run time alignment will pop out errors saying image corrector is not valid(which indicates calibration is not finished yet) or trained data is missing.

The steps of run time are:

1. Load a run time part on platform
2. Acquire image(s) of the part
3. Using Correctors generated from calibration process to transform raw images to corrected images
4. Run vision tasks to find features on that part
5. Calculate the pose differences between golden pose and current pose

6. Motion move to an absolute pose (x, y, Θ) or a relative pose from trained pose(dx, dy, dΘ) provided by vision to bring part to golden pose.

The first three steps of run time process are the same as the steps during the train time process. The differences lie in Step 4-6: In Step 4, there is no need to setup vision task again as it has already been configured in train time and saved in alignment recipe. Step 5: there is no need to change trained golden pose. Instead, that step only uses the trained golden pose to calculate its differences with current run time pose. Step 6: motion device will actually move part to align it to trained golden pose.

There are two types of part handling in Step 6.

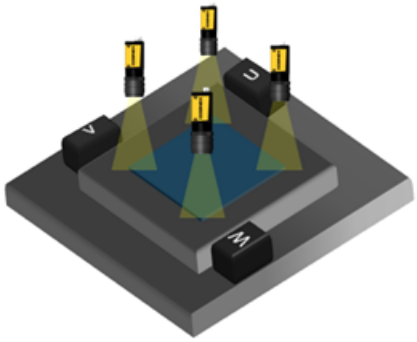
- Align to Base on page 8
- Align to Gripper on page 9

Align to Base

Align to Base: Alignment applications where the part is held to the motion device(such as stage table). During alignment, the application computes motion device parameters that move the run-time part so that it has the same pose as the train-time part. An example of an AlignToBase system is a machine that aligns the cover glass of a phone to a desired orientation for screen printing, where the cover glass rests on a UVW stage.

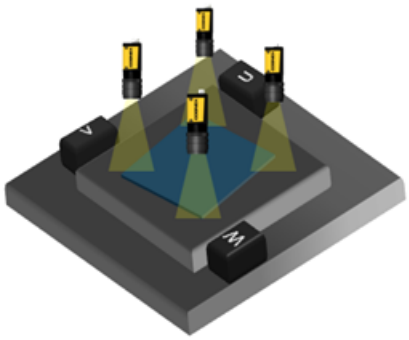
Train time

Golden pose is trained and saved.

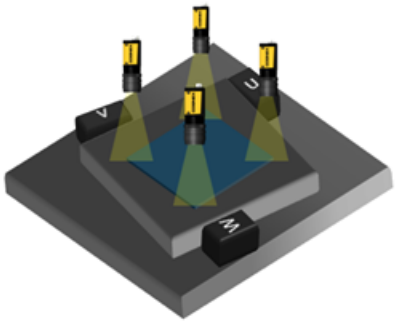


Run time

1. Initial status: Part is placed and attached at a random position on motion device



2. After alignment: motion device moves to new pose which brings the run time part back to desired pose trained during setup.



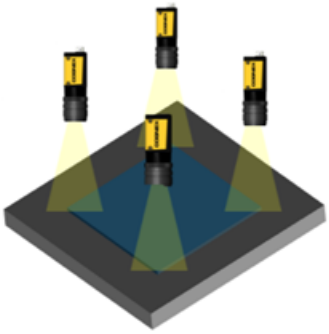
Note: Note that after alignment motion's position might be different with its train time position, but as long as the part itself moves back to its golden pose, then the alignment is finished perfectly.

Align to Gripper

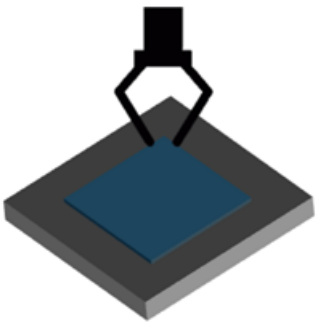
Align to Gripper: Alignment applications wherein the part is placed on a stationary platform waiting for being either picked up by gripper or a processing operation to be carried out. During alignment, the gripper mounted on a robot or the processing unit such as a dispensing head mounted on a gantry adjusts its own position after being directed by the vision system, so that the relationship of the motion device to the run-time part on the platform is the same as one established during a training step.

Train Time

1. Vision train golden pose using features' current locations or other references such as camera centers.

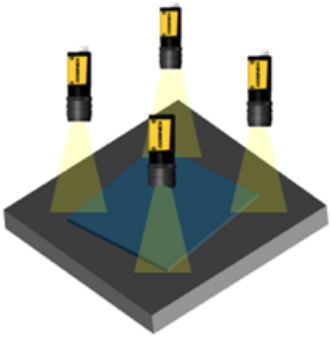


2. Gripper's best position to pick up the part is manually trained and saved by engineer. This pose is also sent to vision for run-time pose computing.

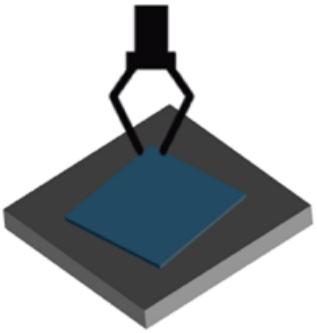


Run Time

1. Vision calculate the features differences between train time and run time and feedback to gripper the new position to pick the run time part.



2. Gripper adjust its pose to fit current part pose, and then pick it up.



Assembly

Assembly Concept

Assembly: Applications where one part needs to be assembled onto another part in a fixed relative pose. In these applications, the poses of both parts are detected by the vision system, and then based upon the assembly requirement, the necessary movements are communicated to the motion system(s).

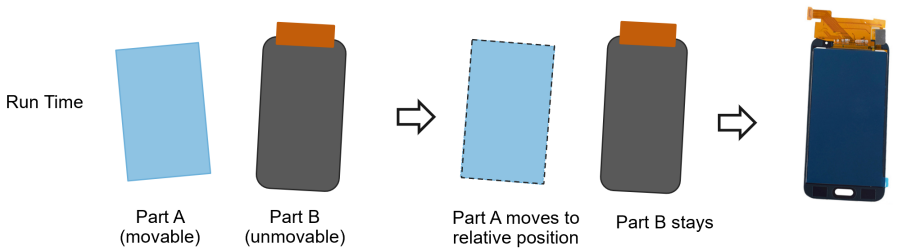
There are two ways to define two parts' relationship when they are assembled: using paired features and using golden pose.

- **Using Paired Features**

  When two parts resemble in profile, and should be assembled center to center, then one of them can directly use the other's features as target pose to form a one-one relationship between features of the two parts. The assembly goal in this mode is to minimize the overall position differences between paired features in a least square sense. Before the pose computation, it is required to select and establish the features that are paired between the two parts.

  Here is an example:

  Part A is a film on stage, Part B is a panel on another stage or a stationary platform. The two parts share similar profiles, and the assembly position is center to center. First, the vision system and motion systems are calibrated, and a common coordinate system is established.



Second, in run time, the vision system detects the feature positions of Part B and Part A. And third, the vision system calculates the position difference between Part A and Part B based on paired features and the compute pose motion device should move to. Following which, the motion system moves either Part A or Part B or both to make them ready to be assembled.

- **Using Golden Pose**

  This mode is used when these two parts to be assembled are quite different from each other that they can not be paired together. Additionally, even if one-on-one correspondences between their features can be established, but the assembly requirements does not center the two parts together, then the 'Golden Pose' mode is employed.

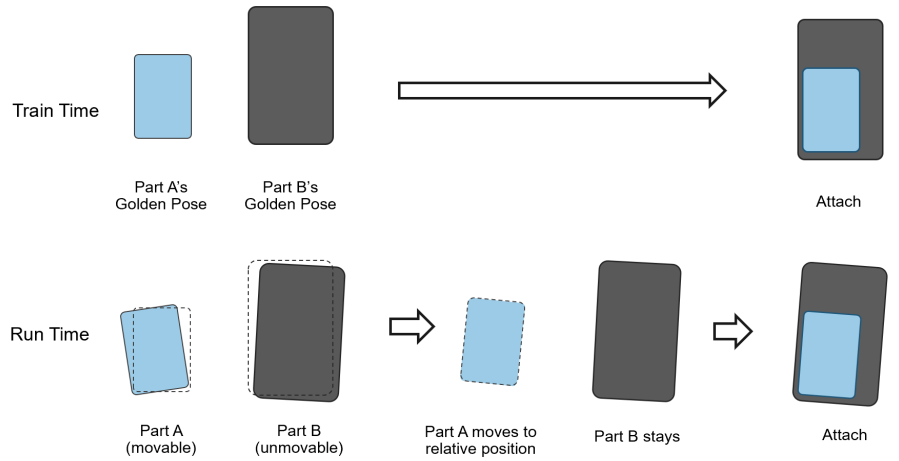
  Here is an example:

  Part A is a battery on stage, Part B is a metal housing on another stage or a stationary platform. Both parts are rectangular, but their final assembly position is not center to center so the mode should be Using Golden Pose.

  First, the vision system and motion systems are calibrated, which establishes a common coordinate system.

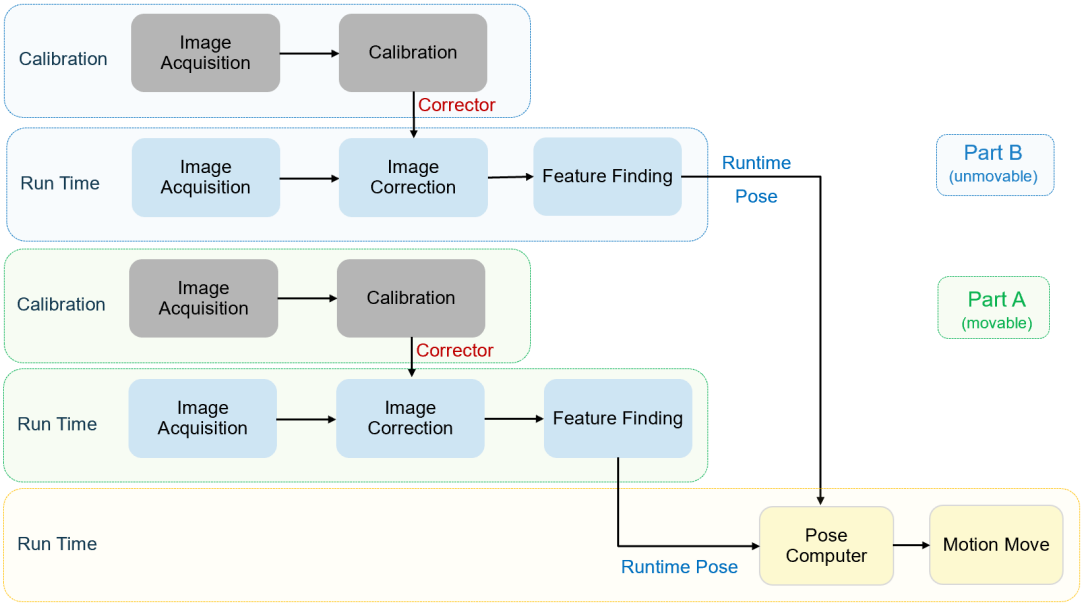
  Then, secondly during train time, both Part A and Part B are placed at desired positions manually, and their features are extracted and trained as golden poses. When they are assembled at their own golden poses, their relative position is exactly as per the application requirements.

  In the third step, during run time, both Part A and Part B's current pose to golden pose differences are calculated by vision system. Following which, the motion system moves either Part A or Part B or both, such that when assembled, Part A and Part B achieve the expected trained relative position.



Assembly Procedure Using Paired Features

When paired features mode is employed, assembly applications have two process steps for each part: (1) Calibration and (2) Run time. During the assembly operation, vision system calculates the position differences between the two parts' features as if one part's position were the other's target pose, and calculates the necessary motion move that will minimize the error across all the paired features and provides the necessary feedback to the motion system.



Calibration

During calibration, both stations run a separate calibration process first, then vision system unifies the two calibration results to create a shared coordinate space for all cameras and motion devices. This shared coordinate space is called Home2D.

After calibration, each camera will have a Corrector which saved the transform between its raw image coordinate space and Home2D space. These Correctors afterward will be used to correct train time and run time images.

Run Time

For the first time of run time tasks running, certain setups should be conducted to configure vision tools for both parts' feature finding:

1. Configure vision tools for Part A and Part B feature finding

2. Use the same feature names for every feature pairs from Part A and Part B so that run time pose computation will recognize them as pairs

3. Save the configurations above in alignment recipe.

After calibration and vision tools configuration, run time tasks are ready to be executed every time when vision system needs to guide motion for completing the assembly task. Run time has two procedures: feature finding and pose computing.

The steps of feature finding that two parts run separately are:

1. Acquire image(s)
2. Using image correctors from calibration data to transform raw images to corrected images
3. Use vision tool to find the features on current part.

The steps of pose computing are:

1. The vision system calculates the pose differences between two parts' paired features.
2. The vision system computes the pose motion device should move based on calculated part pose difference and the mechanical way of part handling.
3. Motion device move to an absolute pose (x,y,theta) or a relative pose(dx,dy,dtheta) provided by vision system from current or trained pose to align part A to fit Pat B.

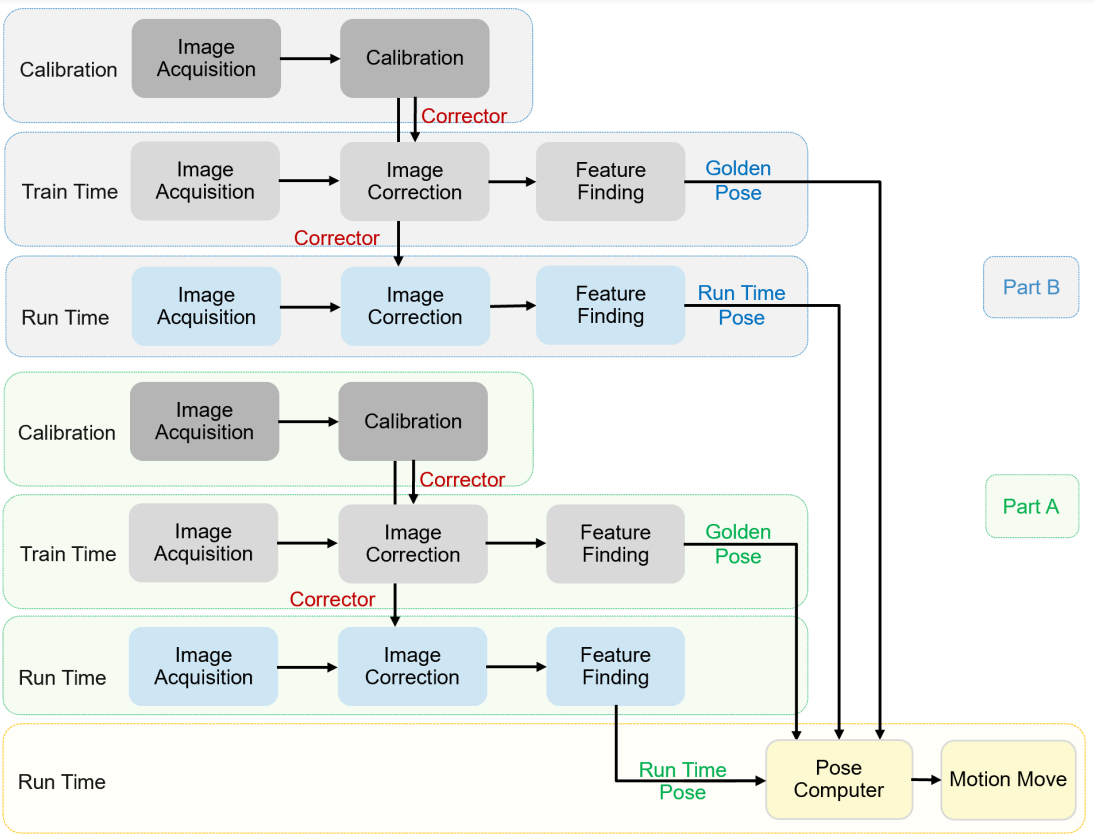
Part B and Part A's feature finding tasks can run asynchronously. However, pose computing should be conducted only after both parts' features are found.

There are three different ways of part handling:

- Assembly Blind Transfer on page 14
- Assembly Guided Pick on page 17
- Assembly Guided Place on page 18

Assembly Procedure Using Golden Pose

When Golden Pose mode is employed, assembly applications have three process steps for each part: (1) Calibration, (2) Train time, and (3) Run time. During the assembly operation, the vision system will take both part's run time pose and golden pose into consideration to calculate a pose one/two stage(s) should move, to bring two parts to desired assembly pose.



Calibration

During calibration, both stations run a separate calibration process first, then vision system unifies the two calibration results to create a shared coordinate space for all cameras and all motion devices. This shared coordinate space is called Home2D.

After calibration, each camera will have a Corrector which saved the transform between its raw image coordinate space and Home2D space. These Correctors afterward will be used to correct train time and run time images.

Train Time

The purpose of train time is to set up a target pose(golden pose) for each part. Golden pose is consisted of a list of features with their coordinates in Home2D.

Each part run the following steps to train golden pose:

1. Manually move a part to the ideal pose with respect to motion device
2. Acquire image(s) of that part

3. Use Correctors generated in calibration process to convert raw images into corrected images
4. Configure vision task to find the features on that part
5. Send command to train golden pose. This golden pose will be saved in recipe.

Run Time

Note: When it comes to run time, Correctors should be available, golden poses are both trained and saved in recipe. If not, run time assembly will pop out errors saying image corrector is not valid(which indicates calibration is not finished yet) or trained data is missing.

Run time is executed every time when vision system needs to guide motion for completing the assembly task, it has two procedures: feature finding and pose computing.

The steps of feature finding that two parts run separately are:

1. Load a run time part on platform
2. Acquire image(s) of the part
3. Using Correctors generated from calibration process to transform raw images to corrected images
4. Run vision tasks to find features on that part

The steps of pose computing are:

1. The vision system computes the difference between each part's run time pose and it target pose.
2. The vision system calculates the one/two motion devices should move to bring Part A to a expected relative pose to Part B based on motion device's current pose or trained pose and different ways of part handling.
3. Motion device(s) moves such that when the two parts are assembled, they are in desired relative pose.

Two parts' feature finding tasks can run asynchronously. However, pose computing should be conducted only after both parts' features are found.

There are three types of machine design of part handling:

- Assembly Blind Transfer on page 14
- Assembly Guided Pick on page 17
- Assembly Guided Place on page 18

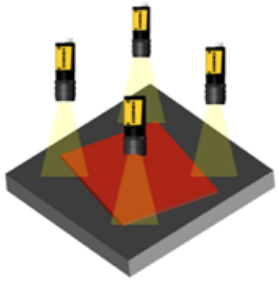
Assembly Blind Transfer

Assembly blind transfer refers to a type of machine design with a fixed part transfer mechanism to assemble two parts. The necessary motion movement is first obtained from the vision system using either Golden Pose or Paired Features assembly modes. Then, the assembly related motion device executes the received feedback. Finally, a separate transfer mechanism will transfer the parts to be assembled. This is illustrated below.

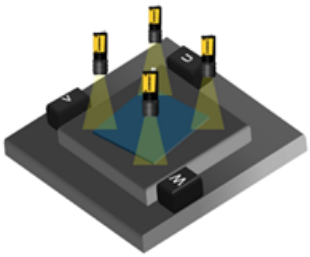
As an example, assume that Part A is placed on a stage that can move Part A in translation(X, Y) as well as rotate Part A(Θ). Also, let's assume that Part B is placed and held securely on a platform and thus is considered to be stationary. A fixed transfer mechanism carries one part to the other.

Blind transfer from stage to stationary platform

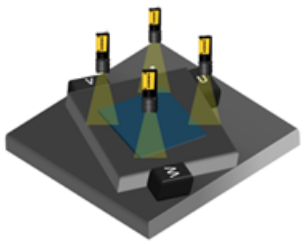
- Step 1: Part B is randomly placed on a stationary platform. Part B's pose is then detected by the vision system.



- Step 2: Part A is randomly placed on stage; Part A’s pose is also detected by the vision system.

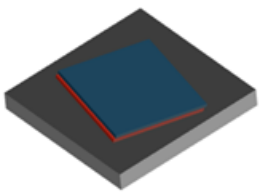


- Step 3: For Paired Features assembly mode, stage adjusts Part A position to fit the detected pose of Part B. For Golden Pose mode assembly applications, the vision system calculates the required motion move using information from Part A and Part B trained golden poses and detected poses.



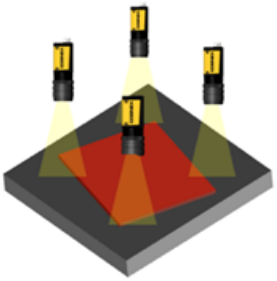
- Step 4: Part A is blindly transferred from stage onto Part B’s stationary platform. The machine is designed such that the transfer mechanism is highly repeatable.

  This process could also be done the opposite: Part B is blindly transferred to Part A’s side.



Blind transfer from gripper to stationary platform

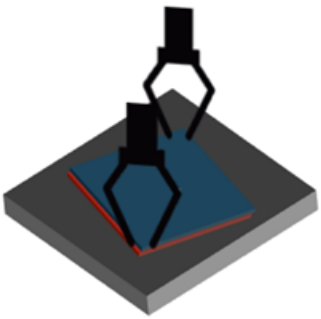
- Step 1: Part B is randomly placed on platform. Part B's pose is then detected by the vision system.



- Step 2: Part A is picked by the gripper in a random pose; Part A’s pose is then detected by the vision system.

- Step 3: For Paired Features assembly mode, the robot adjusts Part A position to fit the detected pose of Part B. For Golden Pose mode assembly applications, the vision system calculates the required motion using information from Part A and Part B trained golden poses and detected poses. The robot then adjusts Part A position based on the feedback from the vision system.

- Step 4: Part A is blindly transferred to Part B’s stationary platform to be assembled onto Part B.

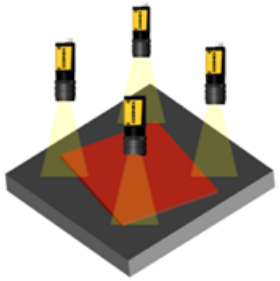


Assembly Guided Pick

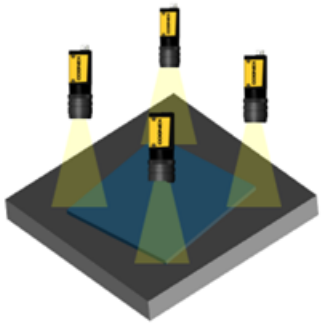
Assembly Guided Pick refers to a type of machine design in which the picking device such as robot adjusts its own position first, then picks up the first part and moves it over to the second part to assembles them together. In this type of design, the motion parameters allow the robot to pick the part at the pick-station correctly at vision system's guidance. After the part is picked, the part is placed on another part at the place-station by moving the robot to a fixed position that is established during training. The part is picked in such a fashion that following assembly the assembled part has the desired assembly characteristics.

This is illustrated below. As an example assume that Part A is placed on a stationary platform. It can be picked up by a gripper which is mounted on a robot end of arm or a gantry. The gripper can move in translation(X, Y) as well as rotate itself (Θ). Also, let's assume that Part B is placed and held securely on a platform and thus is considered to be stationary. Based on the feedback from the vision system, the gripper adjusts its own position first, then picks up Part A and places it on Part B.

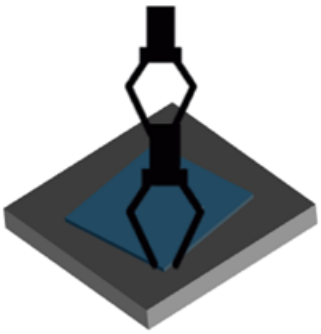
- Step 1: Part B is randomly placed on platform. Part B's pose is then detected by the vision system.



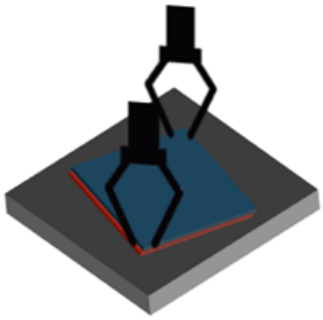
- Step 2: Part A is randomly placed on another stationary platform; Part A’s pose is also detected by the vision system.



- Step 3: Gripper adjust its own position based on vision system feedback to then pick Part A.



- Step 4: The gripper moves Part A over Part B and then places part A onto Part B at a fixed trained position.

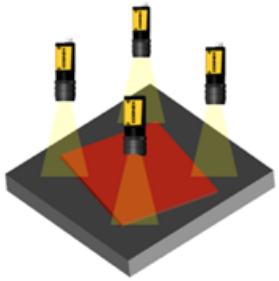


Assembly Guided Place

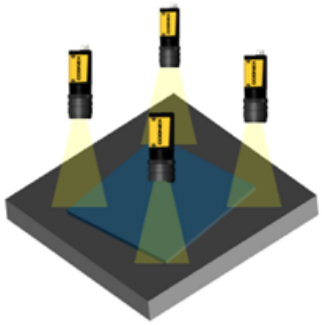
Assembly guided place refers to a type of machine design in which a picking device such as robot picks up one part at a fixed position, and then adjusts its position based on vision system's feedback before or after moving it over to another part's side, and then assemble. The fixed position at pick station is established during train time. At the vision system's guidance, the part is placed on another part at the place-station correctly in such a fashion that the assembled part has the desired assembly characteristics.

- Step 1: Part B is randomly placed on platform; Part B's pose is then detected by the vision system.

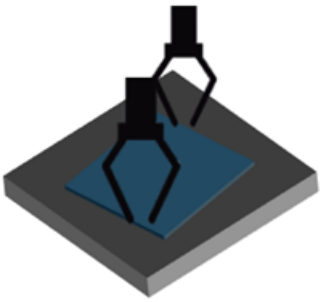
  .



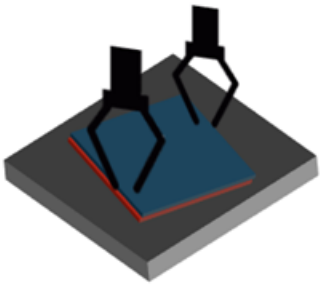
- Step 2: Part A is randomly placed on another platform; Part A’s pose is also detected by the vision system.



- Step 3: Gripper/Robot picks Part A blindly at fixed position.



- Step 4: Using the vision system's feedback from both Part A and Part B, the gripper performs adjustment in its position and then places Part A onto Part B.



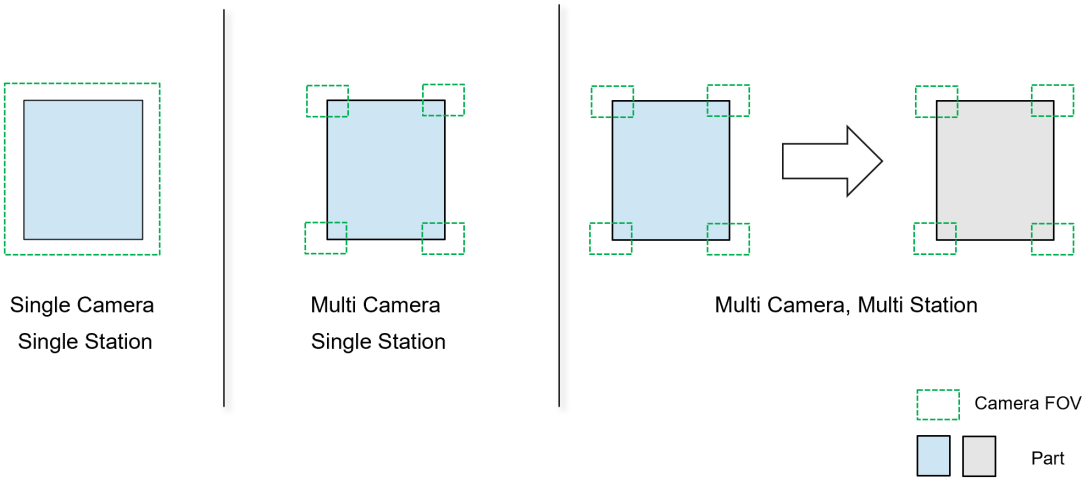
Acquisition

Camera Layout and Installation

Camera Layout

Optical camera layouts are selected based upon the features that are used to detect the part pose, how those features are distributed within the part, their sizes, the expected variation in feature locations and the application accuracy requirements.

Here are some typical camera layouts for single camera station, multiple camera station as well as multiple stations.

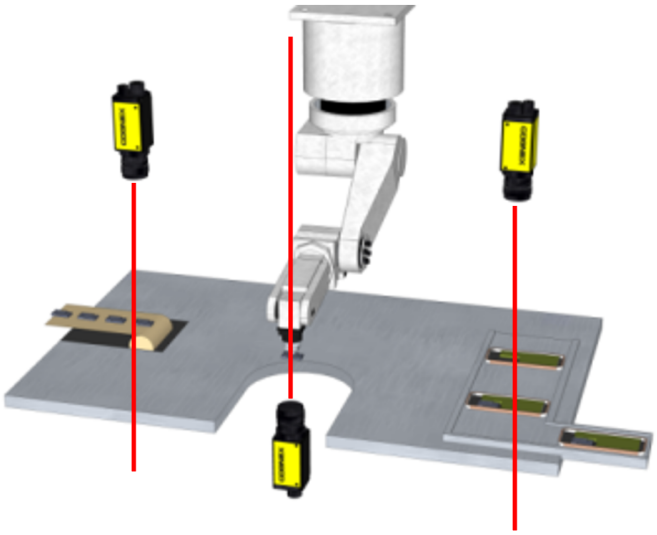


Camera Installation

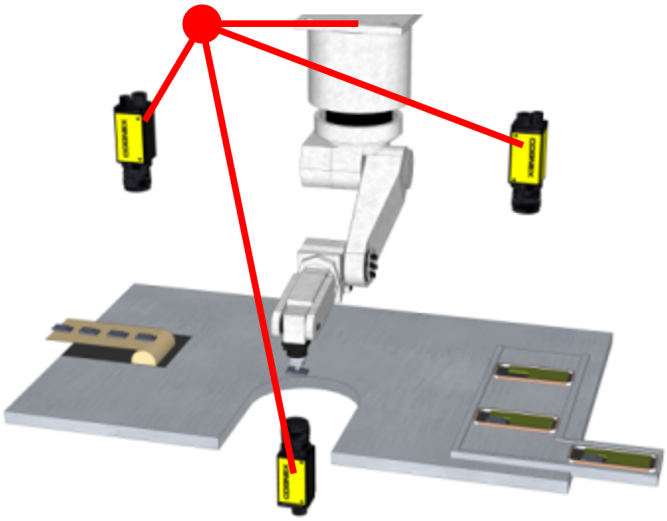
Cognex Checkerboard Calibration on page 51 technology makes it unnecessary to precisely mount cameras.

The following best practices are suggested:

- Camera optical axes be parallel to each other (within ±5°)
- Camera optical axes be perpendicular to the calibration plane (within ±5°)



- Cameras be rigidly mounted to the robot/machine base
- Cameras not be subjected to vibration; avoid mounting vibration sources (fans, motors) to cell



The Following Are Allowed:

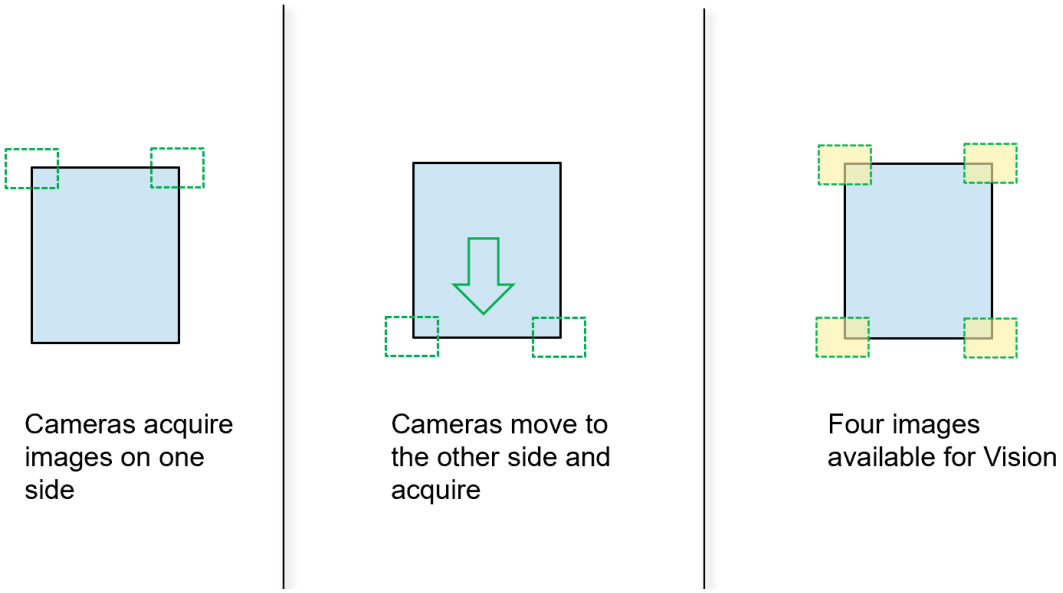
1. Cameras can have different resolutions
2. Cameras can use different lenses
3. Cameras can be mounted at any rotation around the Z-axis
4. Any number of mirrors may be in the optical path

Camera Shuttling or Part Shuttling

In some applications, cameras are shuttled or the parts are shuttled to detect the pose of parts.

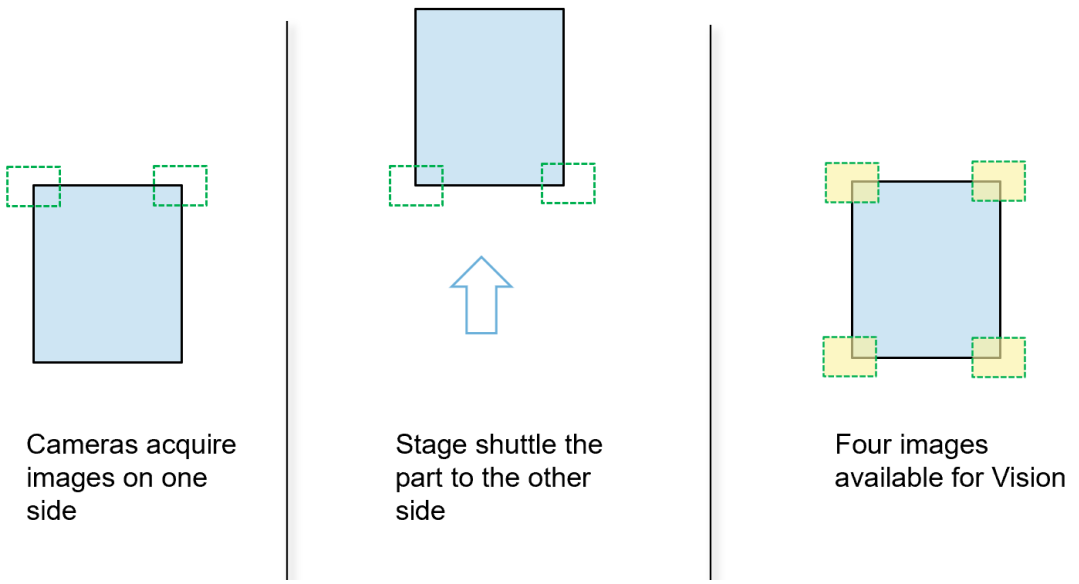
Camera Shuttling

Camera acquires image from one side of the part first, then shuttles to the other side of the part and acquires again. The vision system will combine information from the different images for processing.

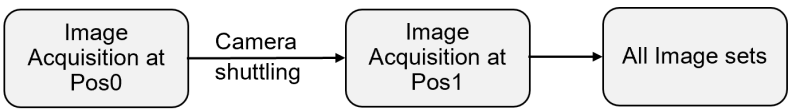


Part Shuttling

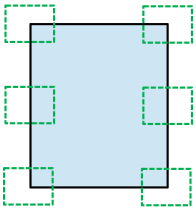
Cameras acquire images on one side of the part, then the part itself moved by platform to the other side under the cameras, then cameras take another set of images. The vision system will combine all information from different images for processing.

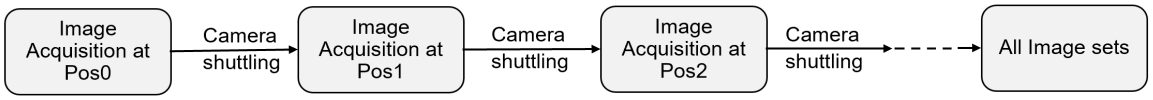


The work flow for vision is as below:



Note that this camera shuttling could be more than 2 positions. AlignPlus supports multiple camera shuttling positions.





Camera Supported

AlignPlus can support up to 16 cameras per IPC, and the camera type that A+ directly supports are:

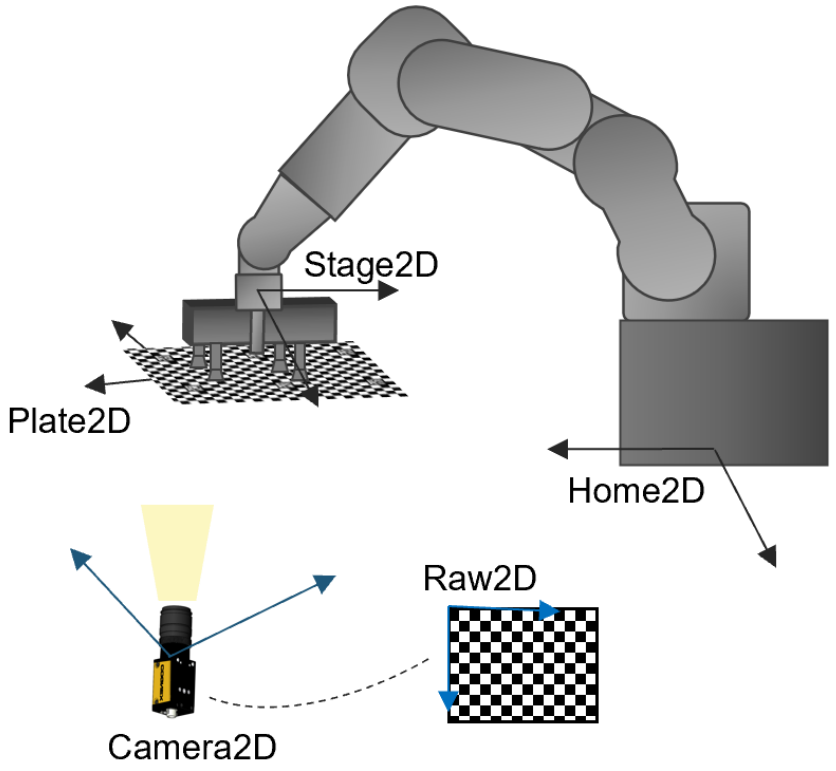
- Cognex 2D Industrial Cameras (CIC cameras)
- Other GigE cameras supported by VisionPro, please refer to VisionPro Camera Support for more information.

For the other type of cameras(such as camera-link, analog camera), user needs to make designer plugin be able to use them in AlignPlus. For more information about Designer Plugin, please refer to **Cognex Designer User Manual/How To... Plugins** and How to add 3rd party acquisition plugin on page 1.

Calibration

Calibration Introduction

The purpose of calibration in alignment/assembly application is to establish a shared coordinate(Home2D) so that all cameras and motion devices' coordinates can be mapped to that shared system. In run time, all feature locations found in camera's image space will be transformed to Home2D coordinates and used for calculating target pose that motion device should move to align/assembly.

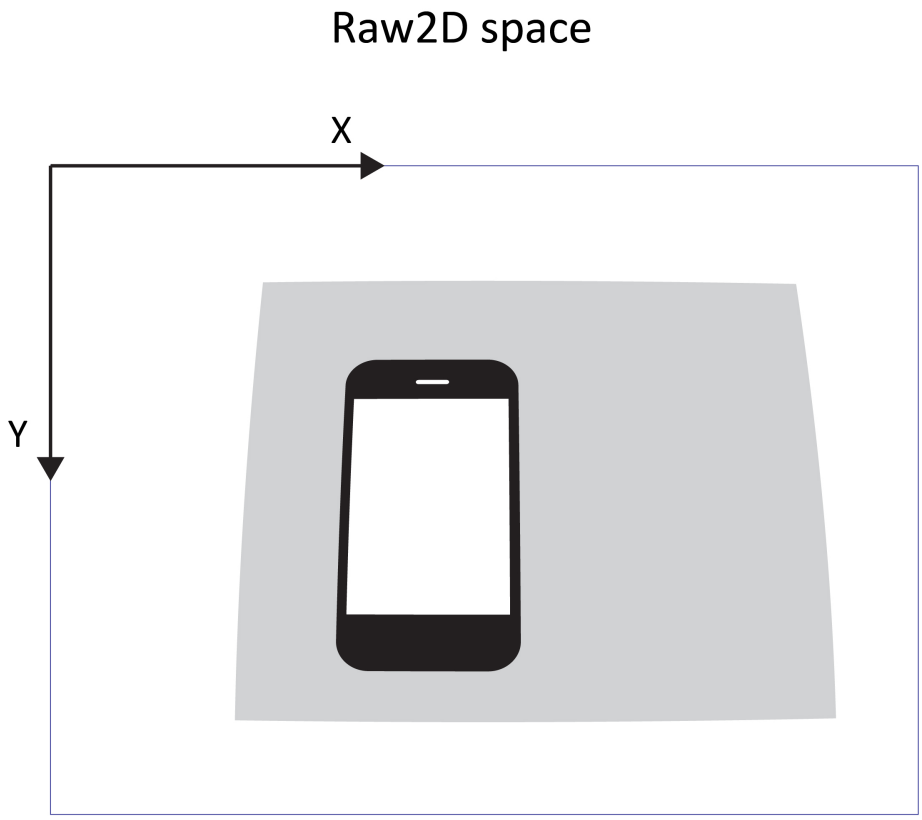


In AlignPlus, there are 6 typical spaces that are used for calibration and run time feature finding: Raw2D, Camera2D, Plate2D, Stage2D, and Home2D.

Coordinate Spaces

Raw2D

Raw2D is the pixel space coordinate system of a single camera. There is one instance of such a coordinate system for each camera. Each camera's Raw2D coordinate system is independent of every other camera.



Acquired image before correction

Camera2D

Camera2D is the physical orthonormal coordinate system for a single camera. There is one instance of such a coordinate system for each camera. The origin of Camera2D is at a position in Home2D that corresponds to the center of the camera's image window. Its X axis is parallel to and points in the same direction as the Raw2D X axis. Its Y axis is perpendicular to the X axis and points in the general direction of the Raw2D Y axis.

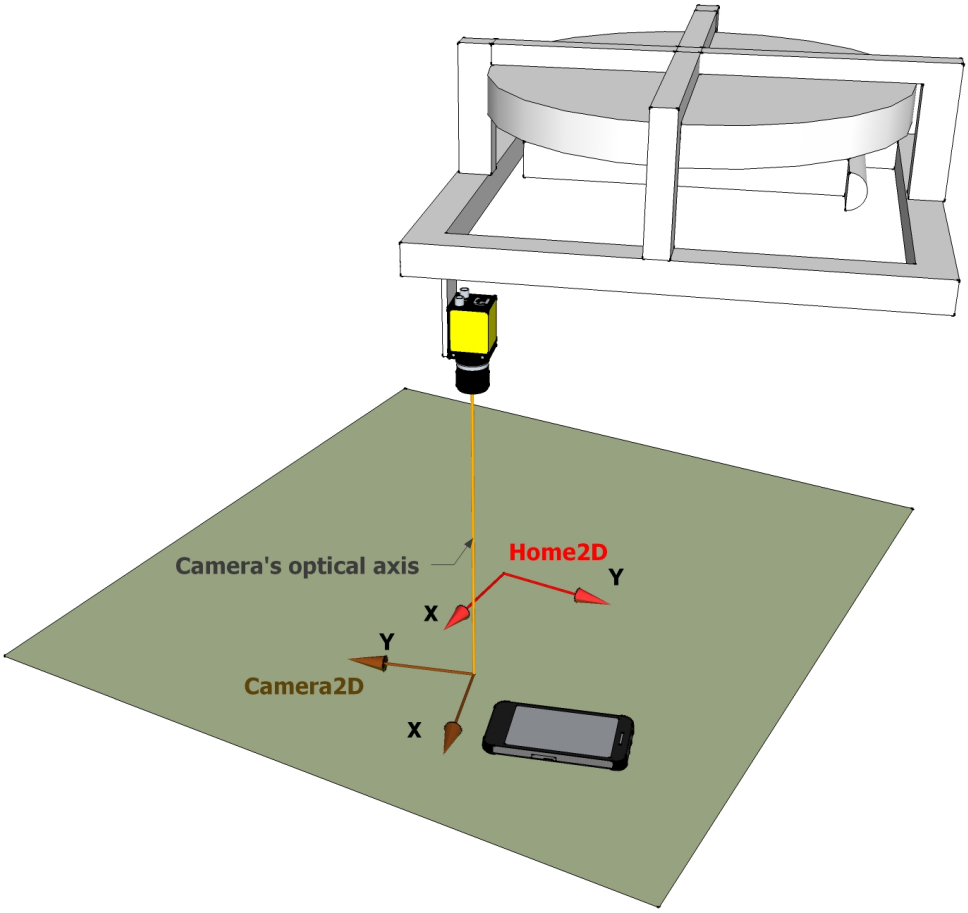
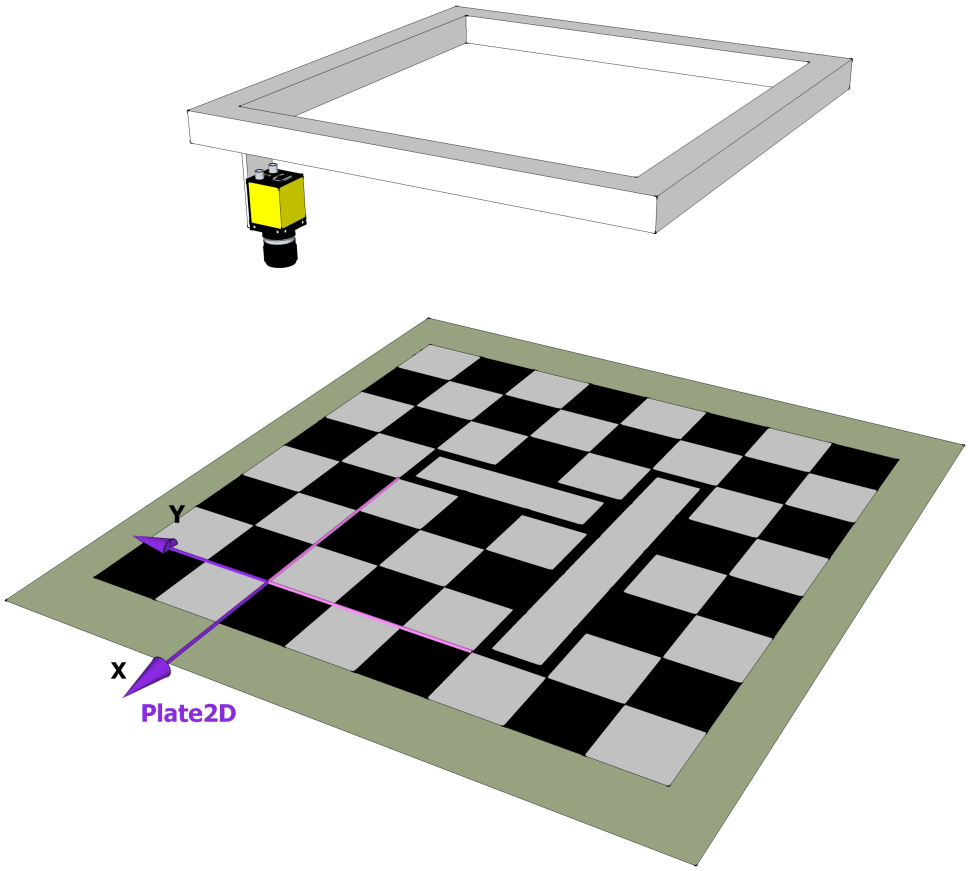


Plate2D

Plate2D is the coordinate system of the calibration plate. At any instance of time, all calibration features viewed by all cameras are described in the same Plate2D coordinate system. However, this space may be moved around in the Home2D coordinates by the motion stage. Plate2D is an orthonormal coordinate system, although its length unit may have a non-identity scale factor from Home2D. Plate2D and Home2D may have different handedness.

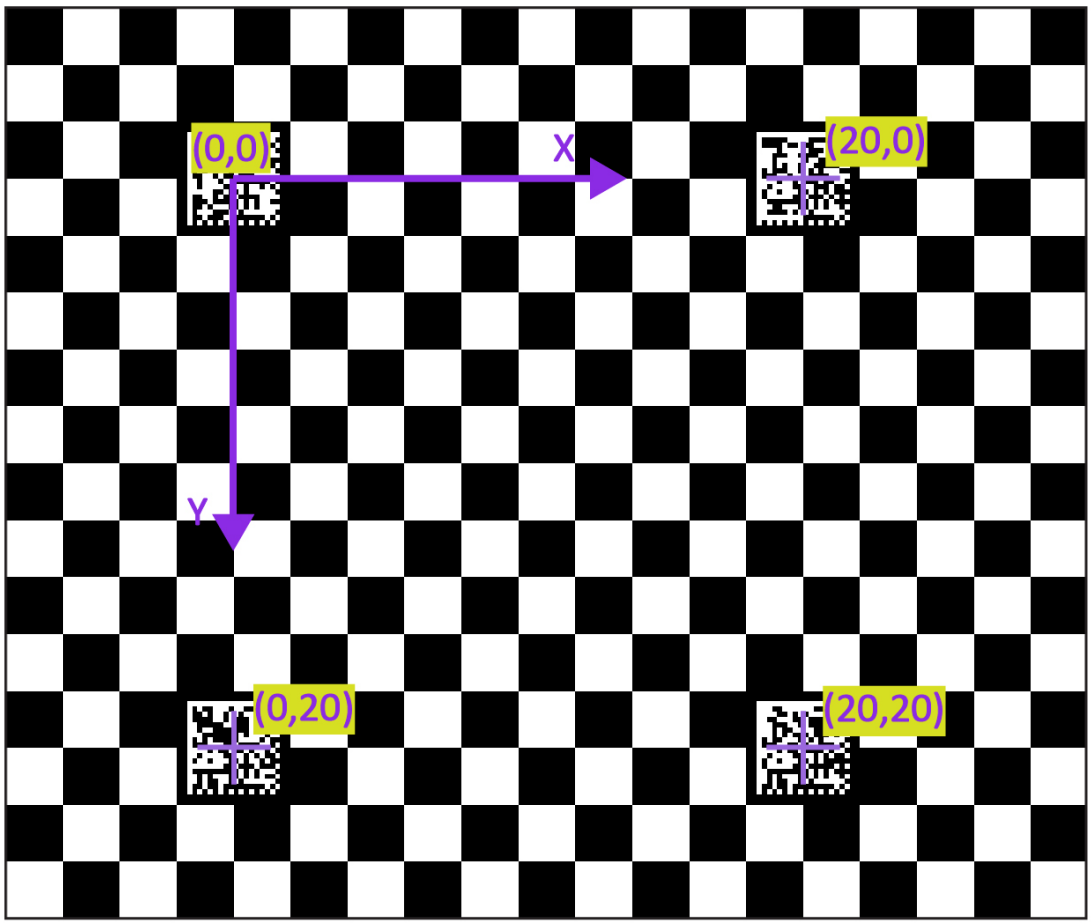
The origin of the Plate2D space is defined either by the fiducial mark or the data matrix codes printed on it. See the documentation on checkerboard calibration for details.

The following figure shows the Plate2D coordinate space in dark purple with pink guides to the fiducial mark features the coordinate space is derived from (in the case of stationary camera configuration).



The following figure shows the Plate2D coordinate space for the previously mentioned checkerboard calibration plate with data matrix codes. The positions marked by the data matrix codes are also shown.

Plate2D space – plate with data matrix codes



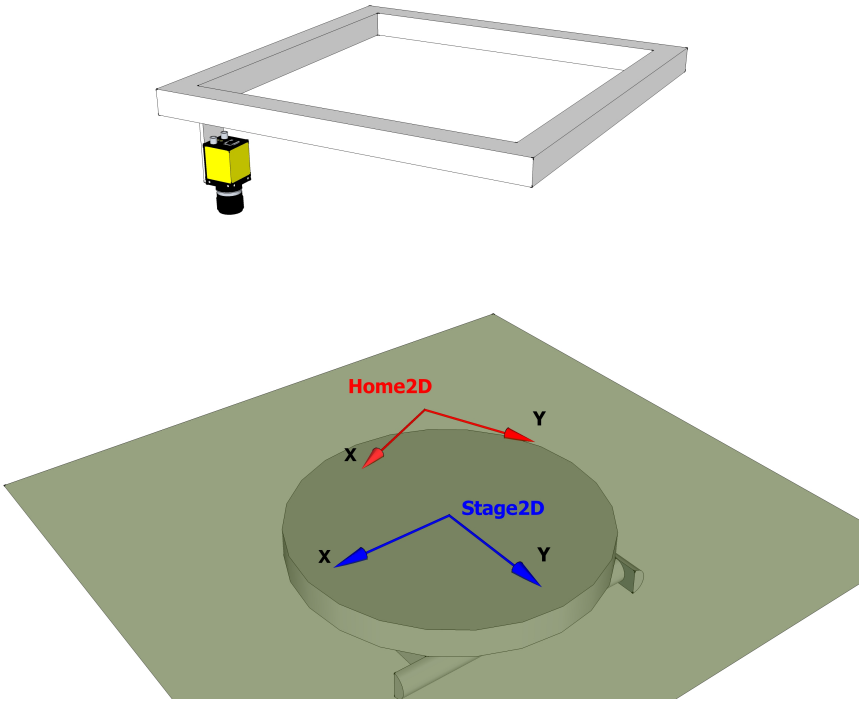
Note:

- As stated earlier, the calibration plate must be parallel to the stage's motion.
- The calibration plate should be in the plane of the part's features that are used for alignment.

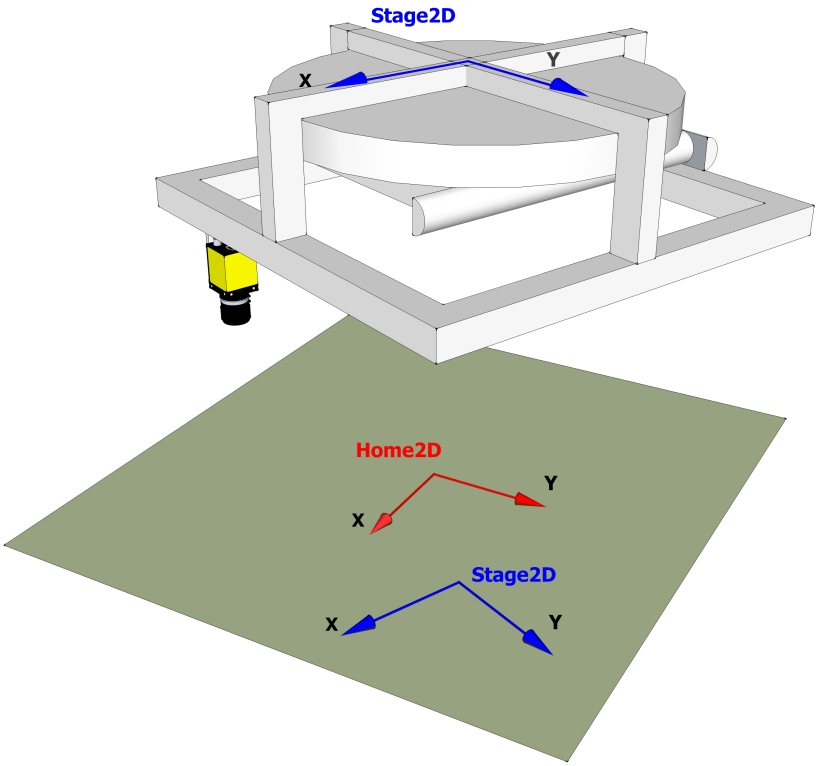
Stage2D

Stage2D is an orthonormal coordinate system that is attached to the motion stage's center of rotation, and moves and rotates along with the motion stage. In the initial position of the stage (that is when X and Y translations, and Theta rotation are zero), Stage2D coincides with Home2D.

The following figure shows the Stage2D coordinate space in blue in the case of stationary camera configuration.



The following figure shows the Stage2D coordinate space in blue in the case of moving camera configuration.

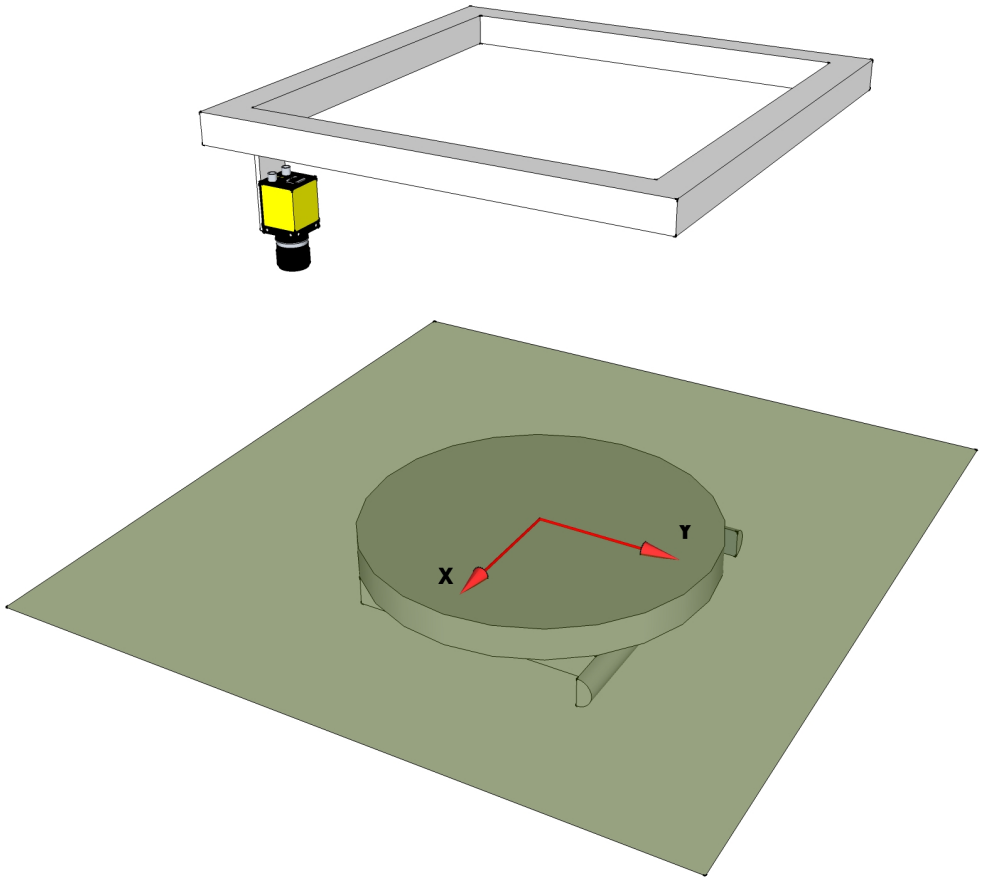


Home2D

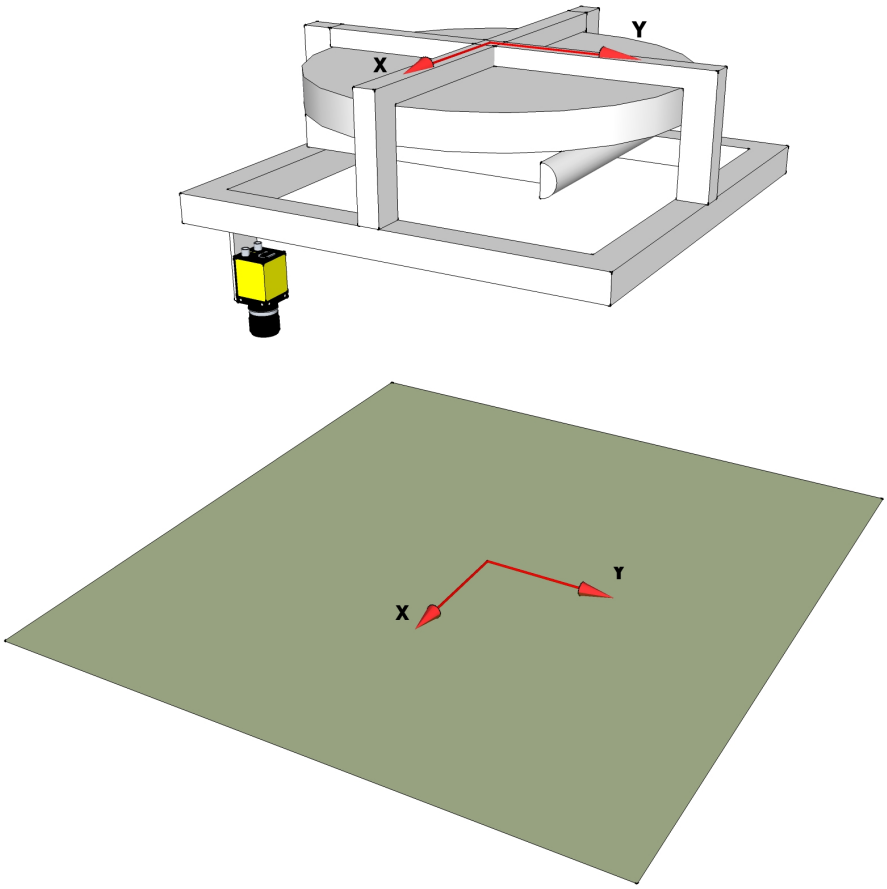
Home2D is the base reference space in which all other coordinate spaces and their relationships are described. Home2D is defined by the initial position of the motion stage; the initial position is the Home position, where (X,Y,Theta) = (0,0,0). Home2D is defined by the X axis of the motion stage and the motion stage's center of rotation. The origin is at the stage's center of rotation when the stage is at the home position. The X axis of Home2D is perfectly aligned with the motion stage's X axis. The Y axis of Home2D is exactly 90 degrees from the X axis and in the general direction of the motion stage's Y axis.

Home2D is the base coordinate space of the motion system.

The following figure shows the Home2D coordinate space in red in the case of stationary camera configuration.



The following figure shows the Home2D coordinate space in red in the case of moving camera configuration.



Handedness of Coordinate Spaces

Independent handedness is assumed among Home2D, Camera2D, and Plate2D. The Home2D handedness is determined by the directions of the motion stage's axes. The Plate2D handedness comes with the calibration plate artwork design. The Camera2D handedness is determined by the number of mirrors in the optical path. These are naturally independent of each other, and are treated as such.

Calibration Types

Hand-eye Calibration Overview

The primary purpose of hand-eye calibration is to calibrate one or more cameras to a motion stage, by establishing a correspondence between features found in images taken by the cameras to the physical coordinates of these features as the motion stage moves to known commanded poses. In addition to other information, the hand-eye calibration results contain a mathematical transform to map image coordinate positions to and from their corresponding coordinates in the coordinate system defined by the motion axes of the stage.

The goal of hand-eye calibration is to obtain accurate estimates for:

- Single-camera single-view calibration model that characterizes the non-linear mapping between image coordinate system and camera coordinate system of each camera.
- Placement pose of each camera.
- Placement pose of the calibration target.
- The mapping from commanded pose to the actual physical pose of the stage in the home coordinate system.

To calibrate, the user must rigidly attach a calibration target to a precision motion stage, e.g. the end-effector of a precision robot arm. The user then commands the motion stage to move the calibration target to a few different commanded poses. It is required that all commanded poses are known. An image of the target is taken by each of the cameras at each such pose. This is called the "stationary-camera configuration".

Alternatively, the cameras may be attached rigidly to the motion stage while the calibration target is fixed. An image of the target is taken by each of the cameras for each pose of the motion stage. This is called the "moving-camera configuration".

When features of the part to be aligned are known in the motion stage coordinate system, the transformation required to align the part are computed in the motion stage coordinate system. Some lower accuracy motion stages may exhibit a skew between the X motion axis and Y motion axis , or a difference in the unit travel between the two axes, or a scale error in the unit travel from the intended nominal value.

The hand-eye calibration process estimates the following systematic errors of a motion stage:

- The magnitude of X unit travel. The direction of the X unit travel is accurate by definition
- The direction of the Y unit travel, i.e., motionYAxisHome2D.angle(). By the definition of Home2D, this should be close to +90 degrees on realistic motion stages
- The magnitude of Y unit travel

During the hand-eye calibration process, AlignPlus recommends that the commanded poses include

- Translation only motion, in the X and Y directions.
- Rotation only motion, with a fixed X and Y value

When a motion device moves in this fashion, AlignPlus produces a calibration report, described in a later section, that contains the scaling and skew information for the motion stage.

How to Move Calibration Target

AlignPlus can generate applications where the application generates the poses during hand-eye calibration. The poses contain the recommended translation only poses and rotation only poses. The poses are determined by the following parameters:

• The maximum and minimum limits for the X axis, and the number of samples when moving between the two limits

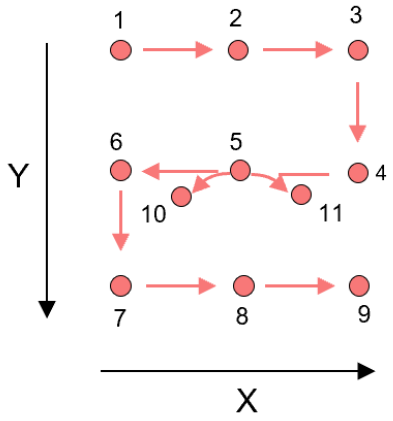
• The maximum and minimum limits for the Y axis, and the number of samples when moving between the two limits

• The maximum and minimum limits for the Theta axis, and the number of samples when moving between the two limits

The table that is shown below shows the default values for the above parameters

	Minimum	Maximum	Steps
X	-2 mm	2 mm	3
Y	-2 mm	2 mm	3
Theta	-1 degree	1 degree	3

The figure below shows the path that will be taken by the motion device for default motion parameters.

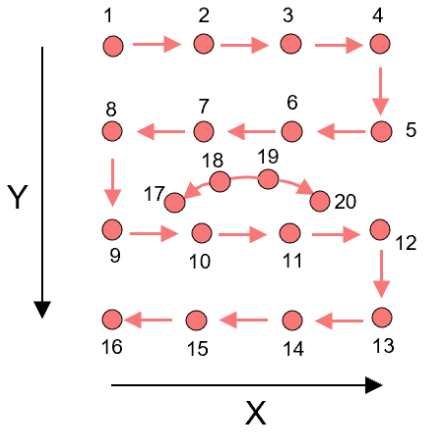


All Points:

Calibration Point	X	Y	Theta
1	-2	-2	0
2	0	-2	0
3	2	-2	0
4	2	0	0
5	0	0	0
6	-2	0	0
7	-2	2	0
8	0	2	0
9	2	2	0
10	0	0	-1
11	0	0	1

Note: The second point of rotation only pose (0,0,0) overlapped with translation only pose 5, so this point is skipped for rotation only poses.

The figure below shows the path what will be taken by motion device for the default limits and increased number of samples.



Calibration Point	X	Y	Theta
1	-2	-2	0
2	-0.667	-2	0
3	0.667	-2	0
4	2	-2	0
5	2	-0.667	0
6	0.667	-0.667	0
7	-0.667	-0.667	0
8	-2	-0.667	0
9	-2	0.667	0
10	-0.667	0.667	0
11	0.667	0.667	0
12	2	0.667	0
13	2	2	0
14	0.667	2	0
15	-0.667	2	0
16	-2	2	0
17	0	0	-1
18	0	0	-0.333
19	0	0	0.333
20	0	0	1

The poses suggested above are all relative poses to the center point (0,0,0) of the moving path. If the center point is not (0,0,0) , all poses' coordinates should be offset based on the center point(x,y,Θ) to obtain the absolute target pose for motion stage to move to. After move is done at each pose, stage' current absolute pose should be sent back to the vision system in return for it to accumulate stage poses which later will be used for hand-eye calibration computation.

If the customer chooses to supply the poses, they should move the motion device as explained above. If this is not feasible, they should generate poses that contain at-least two positions for each degree of freedom for the motion device. That is, two positions for the X, Y and Theta degrees of freedom.

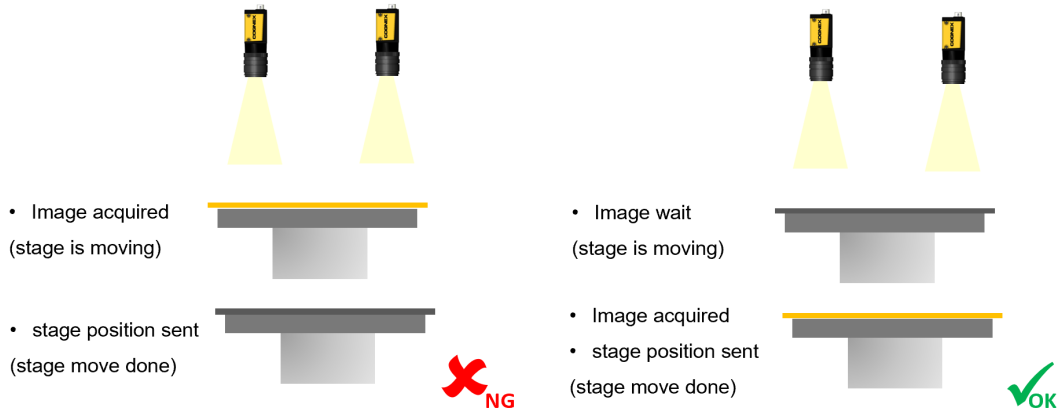
Motion Capability Limitations

After defining the moving path, user need to check whether motion moving range can cover all points on moving path. Note that stage's rotation limit varies when its rotation center changes. For example, stage can rotate between -1 degree to 1 degree when stage x, y coordinates are (0,0), but when stage moves to (0,4), the rotation range might be reduced to -0.8 to 0.6 degrees. Therefore, to check whether all the points are within motion's limits, the best way is to command motion move step by step to each point and see whether the reported actual positions are same as required.

Settling time

During hand-eye calibration, care should be taken to ensure that the motion device has completed motion and has come to a complete rest so that the camera is stationary relative to the calibration target. Failing to do so will affect the quality of the captured image, which in turn affects the hand-eye calibration results. To avoid this situation, a suitable settling time should be added following motion before an image of the target is acquired.

If the customer is responsible for moving the motion device, they should ensure that the target has come to a rest before they send a command to AlignPlus to acquire an image. If the vision application is responsible for moving the motion device, the vision application engineer should apply the settling time in the application.



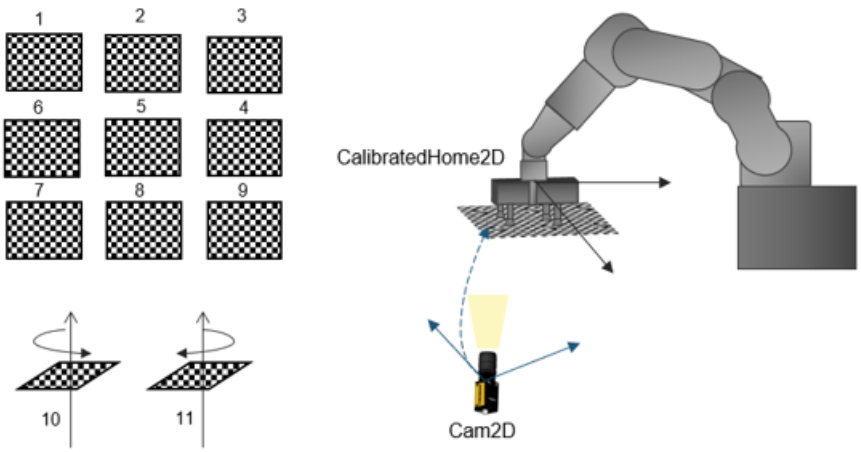
Checkerboard-based Hand-eye Calibration

Checkerboard-based hand-eye calibration involves using a calibration target with a Cognex checkerboard calibration pattern. This pattern contains either an L-Shaped Fiducial, or a set of DataMatrix codes that defines the Plate2D coordinate system the pattern. During hand-eye calibration AlignPlus can locate the corners of pattern in the image and identify their corresponding location Plate2D coordinate system. This correspondence is used to compute the hand-eye calibration results. The advantage of using the checkerboard are

- Each image contains a dense set of features. Therefore, as compared to part-based hand-eye calibration the motion device can move to fewer locations to get a reliable calibration
- If care is taken to ensure that the image of the checkerboard spans the entire field of view of the camera, an accurate estimate of lens and perspective distortion can be obtained
- The user does not have to add any feature finders to extract features in the image. AlignPlus extracts the features from the calibration target
- If an accurate checkerboard is used, the good quality of the calibration, and hence alignment can be obtained even if the motion device suffers from scaling and skew in its linear degrees of freedom

The disadvantages of using a checkerboard as a calibration target are as follows

- On some alignment systems, a special fixture may have to be designed to hold the calibration target. This is because the calibration target should be parallel to the plane of motion of the motion device
- The user would have to take care of the target and ensure that it is not damaged or lost
- Change to light settings, or additional lighting might be necessary to illuminate the calibration target



Part-based Hand-eye Calibration

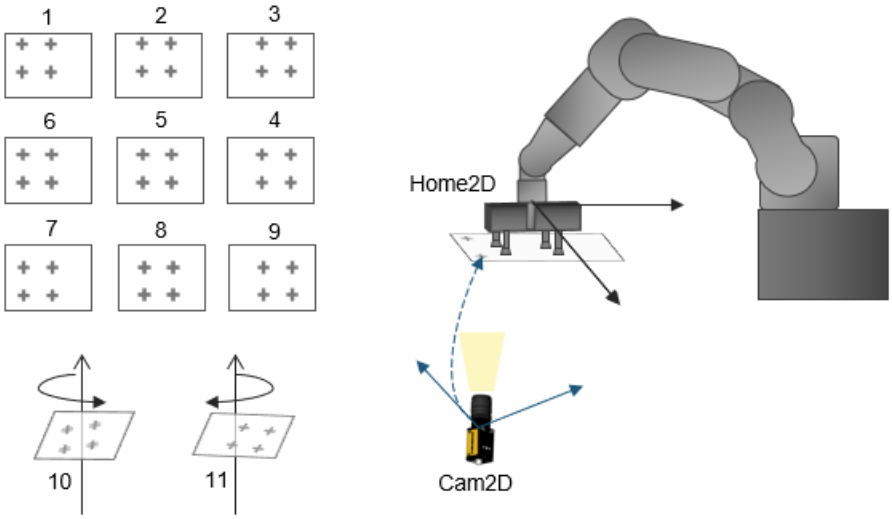
Part-based hand-eye calibration involves using a part that is used during run-time as a calibration target. Part-based hand-eye calibration has the following advantages:

- A special fixture to hold the calibration target is not necessary. This is because the alignment system is designed to hold the part
- Special lighting is not necessary to illuminate the part. The illumination that is used during run-time can be used during calibration.

The disadvantages to using a part for hand-eye calibration are as follows:

- Unlike the checkerboard-based hand-eye calibration, the user will have to add feature finders to locate the features that are used to compute the calibration results
- On systems where lens and perspective distortion must be computed, the calibration target would have to be moved to a lot of positions to generate a dense set of features
- The quality of the hand-eye calibration result and hence alignment is adversely affected by scaling errors in their linear degrees of freedom

The following figure illustrates this with a single camera.



Hand-eye Calibration Result Report

Following hand-eye calibration, AlignPlus generates a hand-eye calibration report. The report contains information about the quality of the captured image and the motion device. The hand-eye calibration is generated for both, checkerboard-based hand-eye calibration and part-based hand-eye calibration. The following subsections describe the salient information contained in this report.

Checkerboard-based Hand-eye Calibration

The following is an example of a calibration report for a hand-eye calibration result, computed for a two, stationary-camera system that has been hand-eye calibrated using a checkerboard. The report contains eight blocks of significance.

• Block 1 contains information about the pixel resolution(PelSize) of the camera,

• Block 2 contains the locations of the camera in Home2D for stationary-camera systems. For moving camera systems, the camera locations are expressed in Stage2D, which is the coordinate system of the gripper.

• Block 3 contains the overall residual error for the hand-eye calibration

• Blocks 7, 8 contain single view residuals for each camera,

• Blocks 4, 5, 6 contain information about the pose of the motion device during the hand-eye calibration process

```
NOTE:
  X & Y values in Raw2D are in pixels.
  X & Y values in Home2D, Stage2D, Plate2D, and Camera2D are in the physical units of your
    cal plate or stage depending on the Calibrator's Home2DUnitLengthReference property.
  Rotations are in degrees.

NumCameras: 2
Type: Stationary camera
NumStagePoses: 11
```

Camera Index	PelSize	Camera Pose in Home2D (Home2DFromCamera2D) X	Y	T degs	Overall Residuals Home2D RMS	Home2D Max	Raw2D RMS	Raw2D Max
0	0.005	56.582	120.625	-179.914	0.004	0.019	0.82	3.83
1	0.005	-4.571	120.538	179.959	0.003	0.015	0.51	3.00

Pose Index	Uncorrected Home2DFromStage2D X	Y	T degs	Corrected Home2DFromStage2D X	Y	T degs	Estimated Home2DFromStage2D X	Y	T degs
0	-2.000	-2.000	0.000	-2.003	-1.992	-0.001	-2.003	-1.992	0.000
1	0.000	-2.000	0.000	-0.002	-1.992	-0.001	0.001	-1.992	0.000
2	2.000	-2.000	0.000	1.998	-1.992	-0.001	1.998	-1.992	-0.001
3	2.000	0.000	0.000	2.000	0.000	-0.001	1.999	0.000	-0.001
4	0.000	0.000	0.000	0.000	0.000	-0.001	0.003	-0.001	0.001
5	-2.000	0.000	0.000	-2.000	0.000	-0.001	-2.005	0.000	-0.003
6	-2.000	2.000	0.000	-1.998	1.992	-0.001	-1.998	1.993	-0.002
7	0.000	2.000	0.000	0.002	1.992	-0.001	0.004	1.992	0.001
8	2.000	2.000	0.000	2.003	1.992	-0.001	2.003	1.992	-0.001
9	0.000	0.000	-1.000	0.000	0.000	0.998	-0.001	0.000	0.995
10	0.000	0.000	1.000	0.000	0.000	-1.000	-0.001	0.000	-1.003

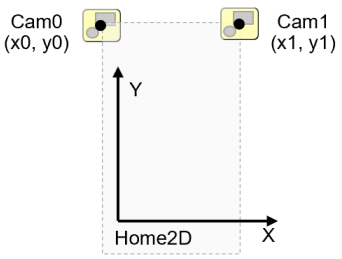
Pose Index	Camera 0 Calibration Residuals Plate2D RMS	Plate2D Max	Raw2D RMS	Raw2D Max	Camera 1 Calibration Residuals Plate2D RMS	Plate2D Max	Raw2D RMS	Raw2D Max
0	0.001	0.003	0.24	0.63	0.001	0.006	0.25	1.28
1	0.001	0.003	0.23	0.66	0.001	0.004	0.27	0.89
2	0.001	0.003	0.24	0.69	0.001	0.004	0.24	0.78
3	0.001	0.004	0.24	0.80	0.001	0.004	0.24	0.81
4	0.001	0.003	0.25	0.66	0.002	0.013	0.32	2.64
5	0.001	0.003	0.25	0.67	0.001	0.004	0.27	0.76
6	0.001	0.007	0.27	1.44	0.001	0.006	0.26	1.26
7	0.002	0.015	0.32	3.04	0.001	0.005	0.24	1.05
8	0.002	0.016	0.34	3.30	0.001	0.005	0.24	1.00
9	0.001	0.004	0.26	0.81	0.001	0.003	0.25	0.64
10	0.001	0.004	0.24	0.72	0.001	0.004	0.25	0.80

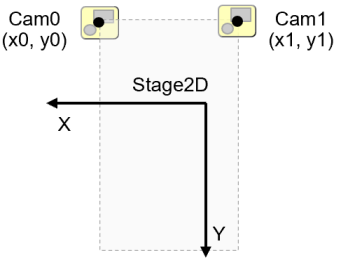
PelSize

The PelSize parameter is a measure of the pixel resolution of the camera, expressed in the measurement unit of the calibration target, which is in millimeters. Pel here stands for pixel and PelSize stands for pixel size. The PelSize parameter is the size of the surface that is imaged by each pixel in the camera. For a camera with square pixels, that captures the image of a surface in a manner that is free of non-affine distortion. PelSize is equal to size of the field of view (FOV) divided by the number of pixels in the camera. For a camera with 1600x1200 pixels, that images a FOV of size 12 x 9 mm, the PelSize is 12mm/1600pixel = 0.0075 mm/pixel.

Camera Pose

The pose of stationary cameras do not change relative to the motion device coordinate system and hence Home2D. Therefore for stationary-camera systems, the camera pose is expressed in Home2D. For moving-camera systems, the camera pose is expressed in Stage2D, as the pose of the cameras are fixed relative to the gripper. This information helps the user to verify if the cameras have been installed corrected.





- Stationary Camera Pose
- Moving Camera Pose

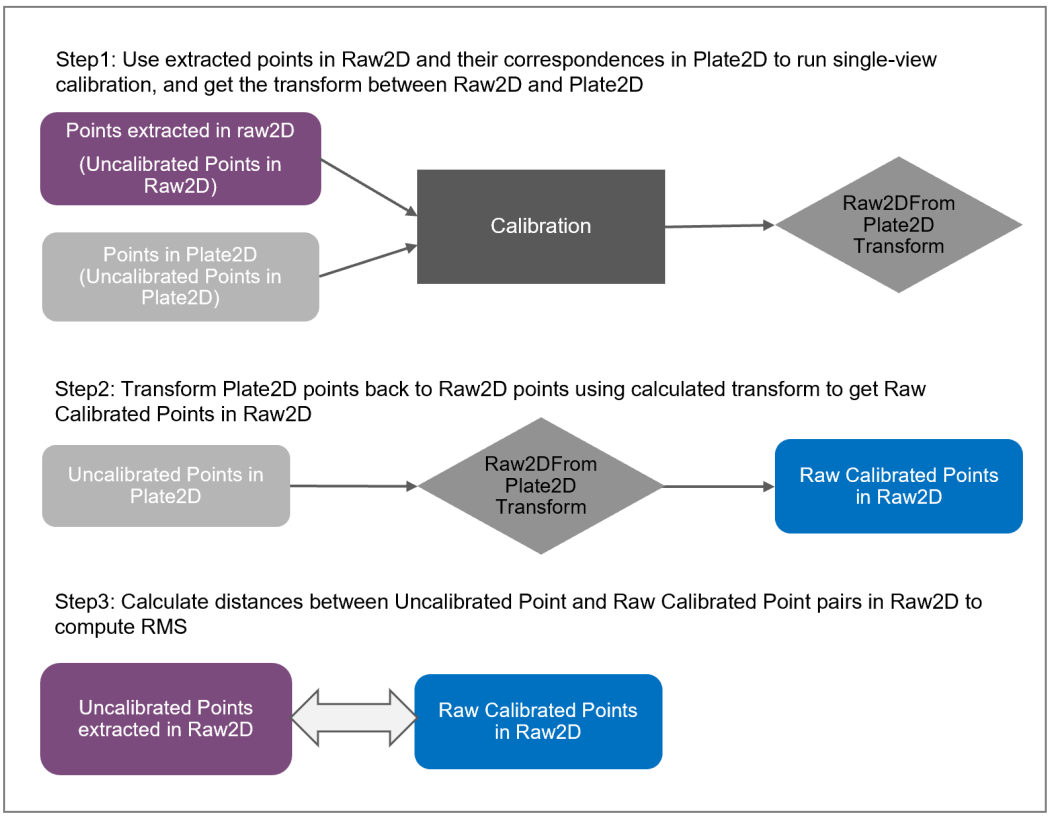
Overall Residuals

The results also include measurement of residual errors, which are indicators of overall quality of the calibration. A good calibration should yield small residual errors. Large residual errors usually indicate bad inputs, examples of which include damaged/inaccurate calibration targets, inaccurate feature extraction, wrong correspondence, inaccurate motion stage, wrong commanded pose, inappropriate lens distortion model, and violations to fundamental requirements.

The residuals are computed in Raw2D and Home2D by mapping the correspondence pairs extracted/computed from all images in the entire hand-eye calibration process. The mapping uses both directions of Raw2DFromHome2D to go to and from Raw2D and Home2D. The unit of the residual errors is pixel in Raw2D and physical units in Home2D. Generally, a residual error of less than 0.25 pixels in Raw2D is considered excellent calibration accuracy.

Single View Residual

Single-view residuals, which are computed from the single-view calibrations for each view of each camera. These residuals are computed in Raw2D and Plate2D, by mapping the correspondence pairs for each view of each camera through the corresponding single-view calibration. Note that the single-view calibrations are not included in the result, only their residuals are. These residuals indicate the quality of the calibration target, feature extractor, correspondence pairs and the selected distortion model. The Raw2D residual errors are in units of pixels, and the Plate2D residual errors are in the physical units of the calibration target. Generally, a residual error of less than 0.1 pixels in Raw2D is considered excellent calibration accuracy.



Uncorrected Home2DFromStage2D

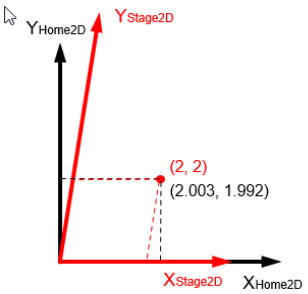
This is the commanded pose of the motion stage. Due to systematic errors in the motion stage, the commanded pose of the motion stage may not match the actual physical pose of the motion stage in Home2D (that is, Home2DFromStage2D). UncorrectedHome2DFromStage2D is used to describe the commanded pose of the motion stage because it represents your best guess of the pose of the motion stage prior to calibration.

Corrected Home2DFromStage2D or **Home2DFromStage2D**

This is the actual physical pose of the motion stage in Home2D. It defines the relationship between Home2D and Stage2D and is represented by a 2D rigid transform. (In the stationary camera configuration, this is the motion of the motion stage of

the object. In the moving camera configuration, this is the motion of the motion stage onto which the cameras are attached – the object platform is stationary in this case.)

Here in this example the UncorrectedHome2DFromStage2D is (2.0, 2.0, 0), but Home2DFromStage2D is (2.003, 1.992).



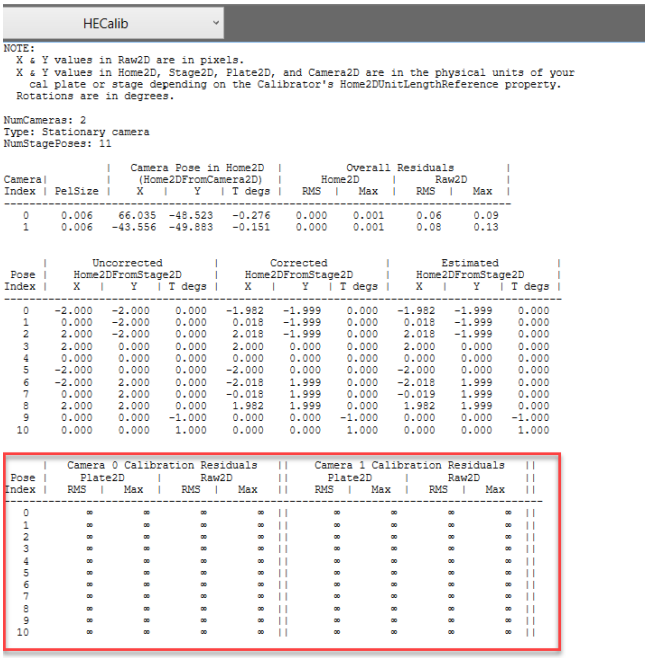
By comparing expected positions and motion's actual positions, hand-eye calibration can calculate the transform between Stage2D and Home2D.

Estimated Home2DFromStage2D

The hand-eye calibration estimates the Home2DFromStage2D poses at which the images were acquired, from which the correspondence data are extracted. These Home2DFromStage2D poses are useful in checking system consistency. When the correspondence data are consistent with the specified input UncorrectedHome2DFromStage2D poses and there are no systematic errors in the stage motion, the estimated Home2DFromStage2D poses should agree closely with the specified input UncorrectedHome2DFromStage2D. Any significant mismatch indicates inconsistency among the input correspondence data and UncorrectedHome2DFromStage2D poses or systematic errors in the motion stage. This is most likely accompanied by large residual errors if the cause of mismatch is inconsistency between the UncorrectedHome2DFromStage2D pose and the input. When such inconsistency occurs, the user should check their system and correct the source of inconsistency, in order to achieve accurate calibration.

Part-based Hand-eye Calibration

The summary of part-based hand-eye calibration is very similar except the for the single view residuals, which are not available in part-based calibration. Typically the part-based calibration process generates a sparse set of features in each view, which makes it infeasible to compute single view residuals.



HECalib

NOTE:
X & Y values in Raw2D are in pixels.
X & Y values in Home2D, Stage2D, Plate2D, and Camera2D are in the physical units of your cal plate or stage depending on the Calibrator's Home2DUnitLengthReference property.
Rotations are in degrees.

NumCameras: 2
Type: Stationary camera
NumStagePoses: 11

Camera Index	PelSize	Camera Pose in Home2D (Home2DFromCamera2D) X	Y	T degs	Overall Residuals Home2D RMS	Home2D Max	Raw2D RMS	Raw2D Max
0	0.006	66.035	-48.523	-0.276	0.000	0.001	0.06	0.09
1	0.006	-43.556	-49.883	-0.151	0.000	0.001	0.08	0.13

Pose Index	Uncorrected Home2DFromStage2D X	Y	T degs	Corrected Home2DFromStage2D X	Y	T degs	Estimated Home2DFromStage2D X	Y	T degs
0	-2.000	-2.000	0.000	-1.982	-1.999	0.000	-1.982	-1.999	0.000
1	0.000	-2.000	0.000	0.018	-1.999	0.000	0.018	-1.999	0.000
2	2.000	-2.000	0.000	2.018	-1.999	0.000	2.018	-1.999	0.000
3	2.000	0.000	0.000	2.000	0.000	0.000	2.000	0.000	0.000
4	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
5	-2.000	0.000	0.000	-2.000	0.000	0.000	-2.000	0.000	0.000
6	-2.000	2.000	0.000	-2.018	1.999	0.000	-2.018	1.999	0.000
7	0.000	2.000	0.000	-0.018	1.999	0.000	-0.019	1.999	0.000
8	2.000	2.000	0.000	1.982	1.999	0.000	1.982	1.999	0.000
9	0.000	0.000	-1.000	0.000	0.000	-1.000	0.000	0.000	-1.000
10	0.000	0.000	1.000	0.000	0.000	1.000	0.000	0.000	1.000

Pose Index	Camera 0 Calibration Residuals Plate2D RMS	Plate2D Max	Raw2D RMS	Raw2D Max	Camera 1 Calibration Residuals Plate2D RMS	Plate2D Max	Raw2D RMS	Raw2D Max
0	∞	∞	∞	∞	∞	∞	∞	∞
1	∞	∞	∞	∞	∞	∞	∞	∞
2	∞	∞	∞	∞	∞	∞	∞	∞
3	∞	∞	∞	∞	∞	∞	∞	∞
4	∞	∞	∞	∞	∞	∞	∞	∞
5	∞	∞	∞	∞	∞	∞	∞	∞
6	∞	∞	∞	∞	∞	∞	∞	∞
7	∞	∞	∞	∞	∞	∞	∞	∞
8	∞	∞	∞	∞	∞	∞	∞	∞
9	∞	∞	∞	∞	∞	∞	∞	∞
10	∞	∞	∞	∞	∞	∞	∞	∞

Stage Validation Result

The purpose of stage validation is to verify that the stage moves to its commanded poses (X, Y, Theta), and to characterize certain types of systematic errors in the observed motion.

Like the hand-eye calibration usage model, this tool uses correspondence data extracted from all views (images) from all cameras, along with the UncorrectedHome2DFromStage2D pose associated with each view. The tool validates the motion stage for all sets of UncorrectedHome2DFromStage2D poses that contain motion corresponding to the metric requested for in the input parameters. The following metrics are available:

- XScale
- YScale
- ThetaScale
- Skew between the X and Y motion axes

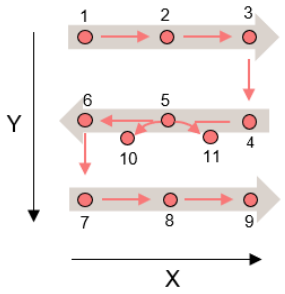
During stage validation AlignPlus requires that subsets of UncorrectedHome2DFromStage2D poses contain elements where all degrees of freedom except for one are constant. For example, a subset of commanded poses should contain constant Y and Theta values. This would allow validation of the X degree of freedom.

On AlignPlus applications where the vision system controls the motion device, the stage validation is performed as a part of the hand-eye calibration process. AlignPlus moves the stage in a fashion that makes stage validation feasible. For application where the PLC controls the motion device it is recommended that the stage is moved in a fashion recommended in this document

```
NOTE:
  Residual Values
    - X and Y residual values are in the physical units of your cal plate or
      your stage depending on the Calibrator's Home2DUnitLengthReference property.
    - Theta residual values are in degrees.
  Skew is in degrees.

1
| X Scale | X Residuals | | Y Residuals | | T Residuals | | |
|         | RMS   | Max   | RMS   | Max   | RMS   | Max   | Poses
|  1.0006 | 0.002 | 0.002 | 0.005 | 0.006 | 0.001 | 0.001 | 0,1,2
|  0.9998 | 0.007 | 0.010 | 0.013 | 0.017 | 0.002 | 0.002 | 3,4,5
|  0.9998 | 0.007 | 0.010 | 0.010 | 0.014 | 0.001 | 0.002 | 6,7,8

2
| Y Scale | X Residuals | | Y Residuals | | T Residuals | | |
|         | RMS   | Max   | RMS   | Max   | RMS   | Max   | Poses
|  0.9961 | 0.004 | 0.005 | 0.008 | 0.011 | 0.001 | 0.001 | 0,5,6
|  0.9960 | 0.002 | 0.003 | 0.003 | 0.004 | 0.000 | 0.001 | 1,4,7
|  0.9960 | 0.001 | 0.002 | 0.001 | 0.001 | 0.000 | 0.000 | 2,3,8

3
| T Scale | X Residuals | | Y Residuals | | T Residuals | | |
|         | RMS   | Max   | RMS   | Max   | RMS   | Max   | Poses
|  0.9990 | 0.002 | 0.003 | 0.001 | 0.001 | 0.002 | 0.004 | 4,9,10

4
| Skew(degs) | X Residuals | | Y Residuals | | T Residuals | | |
|            | RMS   | Max   | RMS   | Max   | RMS   | Max   | Poses
|  -0.0697   | 0.006 | 0.010 | 0.010 | 0.018 | 0.001 | 0.002 | 0,1,2,3,4,5,6,7,8
```

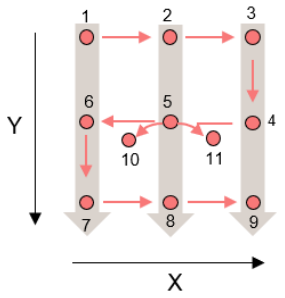
1. **XScale**

   Stage's x scale based on three movements along x axis, where the scale is the ratio of translations in the x direction for UnCorrectedHome2DFromStage2D and CorrectedHome2DFromStage2D. The closer it is to 1, the better.



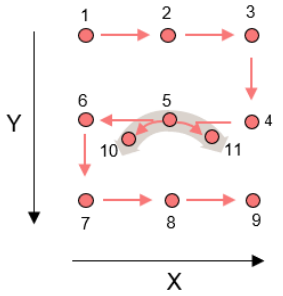
2. **YScale**

   Stage's y scale based on three movements along y axis, where the scale is the ratio of translations in the y direction for UnCorrectedHome2DFromStage2D and CorrectedHome2DFromStage2D. The closer it is to 1, the better.



3. **Theta Scale**

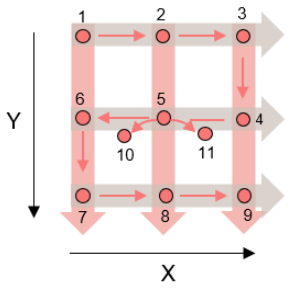
   Stage's theta scale based on rotation movement, where the scale is the ratio of rotation for UnCorrectedHome2DFromStage2D and CorrectedHome2DFromStage2D. The closer it is to 1, the better.



4. **Skew**

   Skew between the X and Y motion axes based on all movements along x and y axes.

   X axis and Y axis are perpendicular if skew is 0.



Cross Calibration

AlignPlus performs alignment of parts using motion devices by manipulating the parts so that the relative location of their features satisfies a desired geometric constraint. This is made possible by defining the features in a common coordinate system. The coordinate system of choice is Home2D.

On some alignment systems, the parts might be in different substations with each part imaged by a unique set of cameras. One would have to hand-eye calibrate all the cameras in order to establish the relationship between Raw2D and Home2D. Due to the way the part handling system is built it might not be possible to run hand-eye calibration in some of the substations. Some of the reasons could be

- There might be insufficient space to move the calibration target during hand-eye calibration process at these substations
- Hand-eye calibration could be time consuming, especially when a customer uses a part to perform hand-eye calibration, where the customer must move the calibration target to many positions for accurate calibration. The customer might not want run multiple hand-eye calibration operations at each substation.
- On some alignment systems, the parts may be transferred by a motion device that is different than the motion device used for alignment. An example would be a bonding system that might use a motion device to align the parts but might use a repeatable gantry that shuttles between two positions to assemble the aligned parts.

There are a few ways in which cameras can be calibrated to map image features to Home2D. A couple of techniques are manual calibration and cross-station calibration. This document describes cross-station calibration, also known sometimes as remote station calibration.

Cross-station calibration uses the result of hand-eye calibration that has been performed on one of the substations to compute the relationship between Raw2D and Home2D for the cameras at the other substations. AlignPlus implements

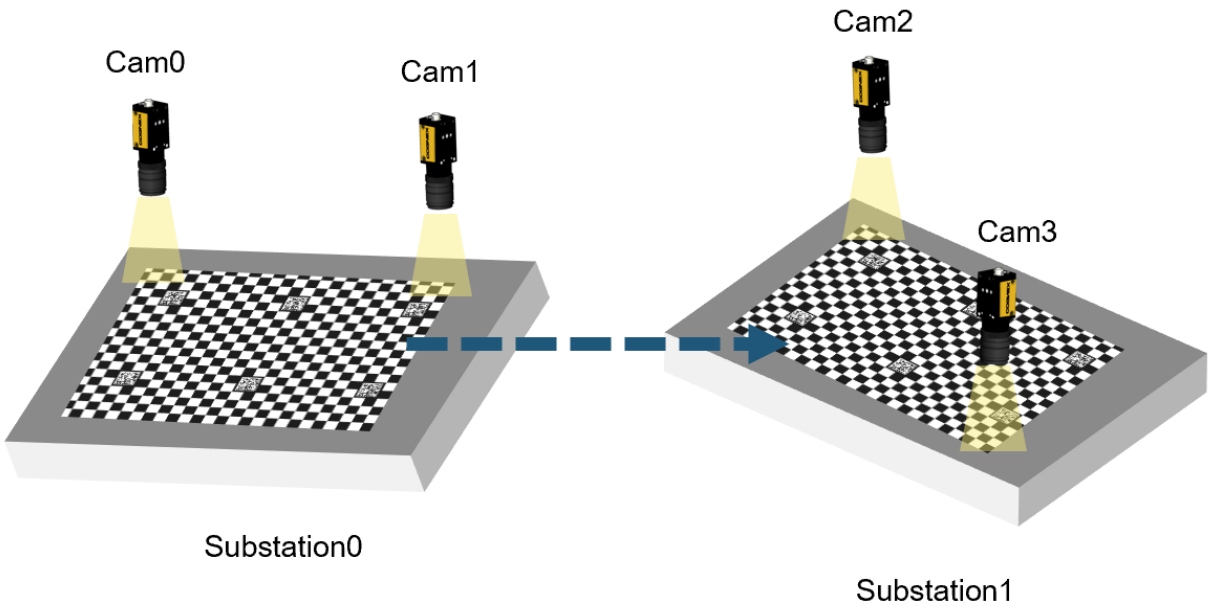
three techniques to perform cross-station calibration. All the techniques compute the relationship between the gripper, Stage2D, and a calibration target used during cross-station calibration and they assume that this relationship does not change when station is cross-station calibrated with the substation that has been hand-eye calibrated.

AlignPlus supports three types of cross-calibration:

- Checkerboard-based Cross Calibration on page 43
- Part-based Cross Calibration on page 44
- Hybrid Cross Calibration on page 46

Checkerboard-based Cross Calibration

This mode uses a checkerboard as the calibration target.

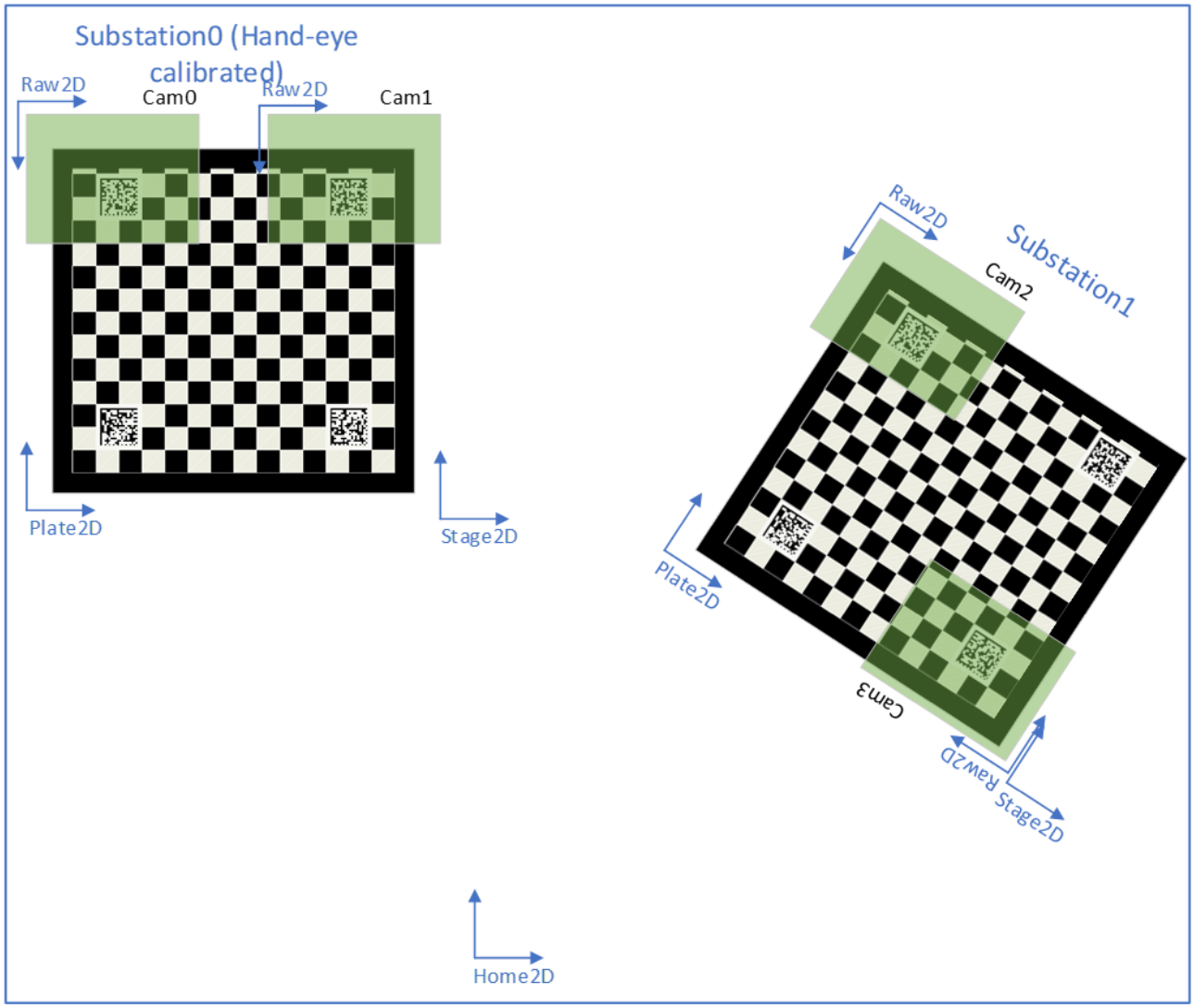


The advantage of using the checkerboard are:

- If care is taken to ensure that the image of the checkerboard spans the entire field of view of the camera, an accurate estimate of lens and perspective distortion can be obtained
- The user does not have to add any feature finders to extract features in the image. AlignPlus extracts the features from the calibration target

The disadvantages of using a checkerboard as a calibration target are as follows

- On some alignment systems, a special fixture may have to be designed to hold the calibration target. This is because the calibration target should be parallel to the plane of motion of the motion device
- The user would have to take care of the target and ensure that it is not damaged or lost
- Change to light settings, or additional lighting might be necessary to illuminate the calibration target



Cam2 and Cam3 are cross-station calibrated to SubStation0 using a checkerboard as a calibration target

Sequence of Commands

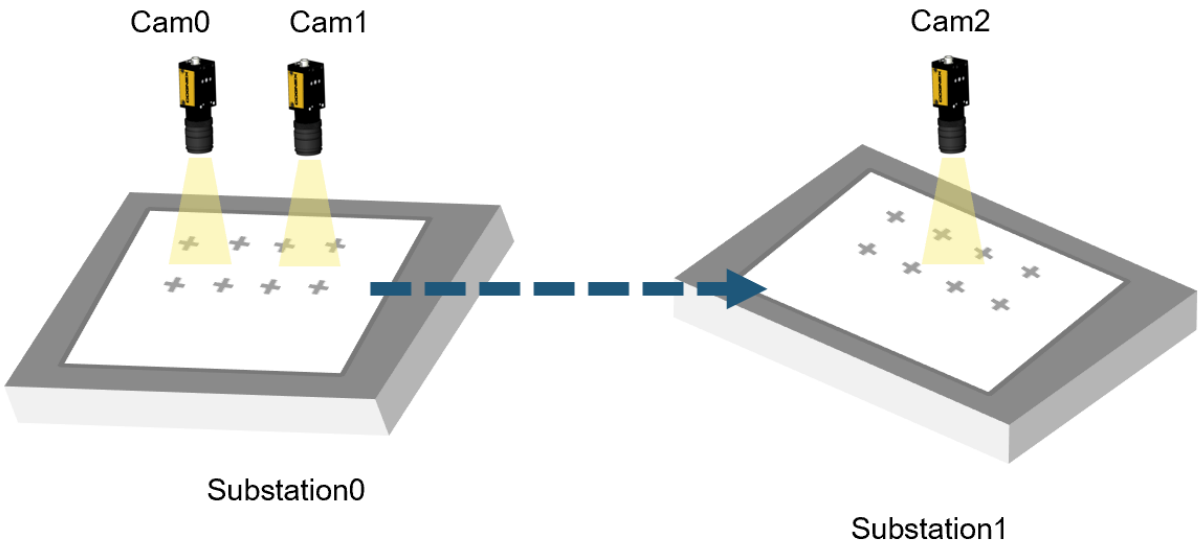
The following are the sequence of commands that must be sent from the PLC to AlignPlus to perform cross-station calibration for the above system

1. The gripper should present the checkerboard target to the substation that has been hand-eye calibrated, SubStation0 in the figure above, and should send the calibration command along with gripper pose at this station.

2. The gripper should present the checkerboard target to the substation that must be cross-calibrated, SubStation1 in the figure above, and should send the calibration command along with gripper pose at this station. Care should be taken to ensure that Plate2DFromStage2D is constant during the entire calibration process, which means that the relationship between the calibration target and the gripper should not change.

Part-based Cross Calibration

This mode uses a part as the calibration target. This technique is used in applications where the gripper can transfer only the run-time part and cannot be adapted to transfer a checkerboard.

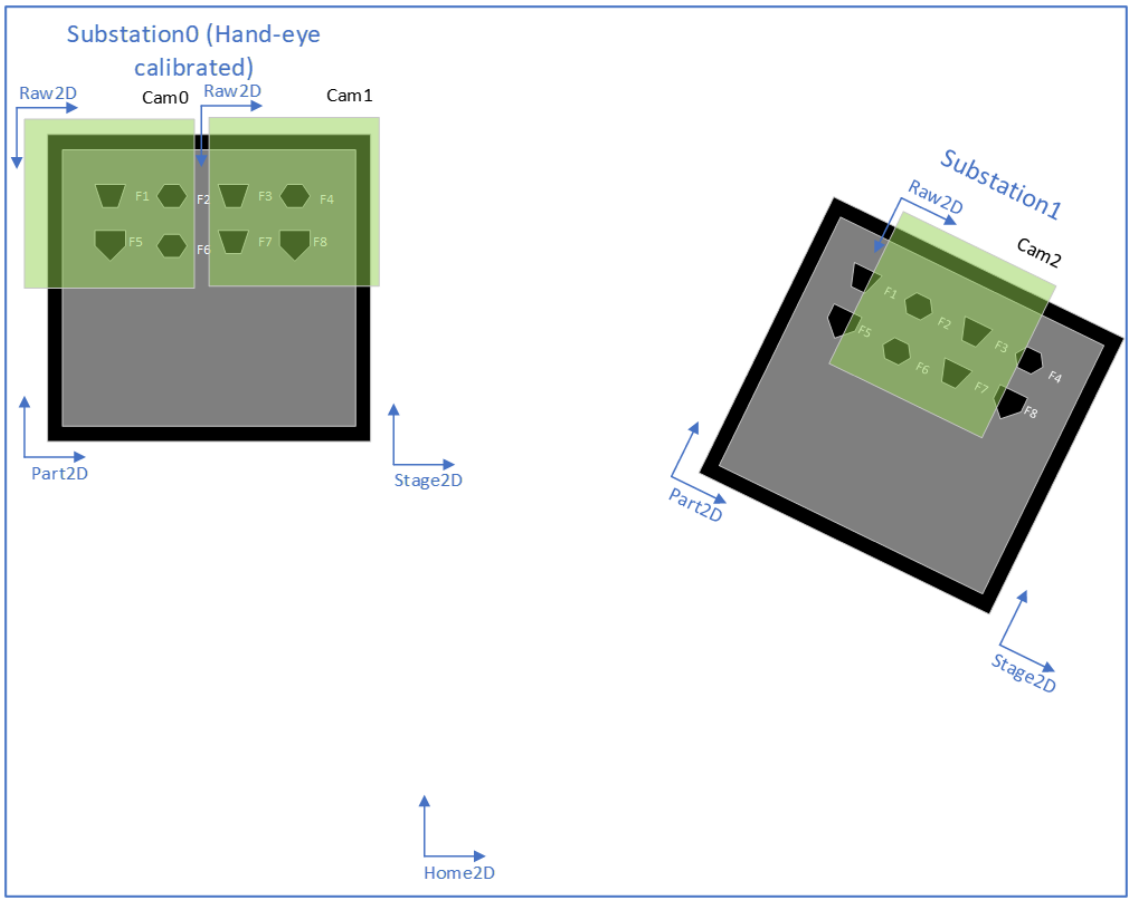


Part-based cross calibration has the following advantages

- A special fixture to hold the calibration target is not necessary. This is because the alignment system is designed to hold the part
- Special lighting is not necessary to illuminate the part. The illumination that is used during run-time can be used during calibration.

The disadvantages to using a part for hand-eye calibration are as follows

- Unlike the checkboard-based cross calibration, the user will have to add feature finders to locate the features that are used to compute the calibration results
- It is difficult to use this mode of calibration to compute lens and perspective distortion as it is not practical to add many feature finders that would be needed for computation.
- Each camera in SubStation1 should see at least three features which have already been located in SubStation0 for successful calibration, this may be hard to achieve when there are very few features on run-time part.



Cam2 in SubStation1 is cross-calibrated to SubStation0 that has been hand-eye calibrated. Note that even though Cam2 locates some features from Cam0 and other features from Cam1, the calibration would still succeed because the locations of all the features are known in Home2D in both the substations.

Sequence of Commands

The following are the sequence of commands that must be sent from the PLC to AlignPlus to perform cross-station calibration for the above system

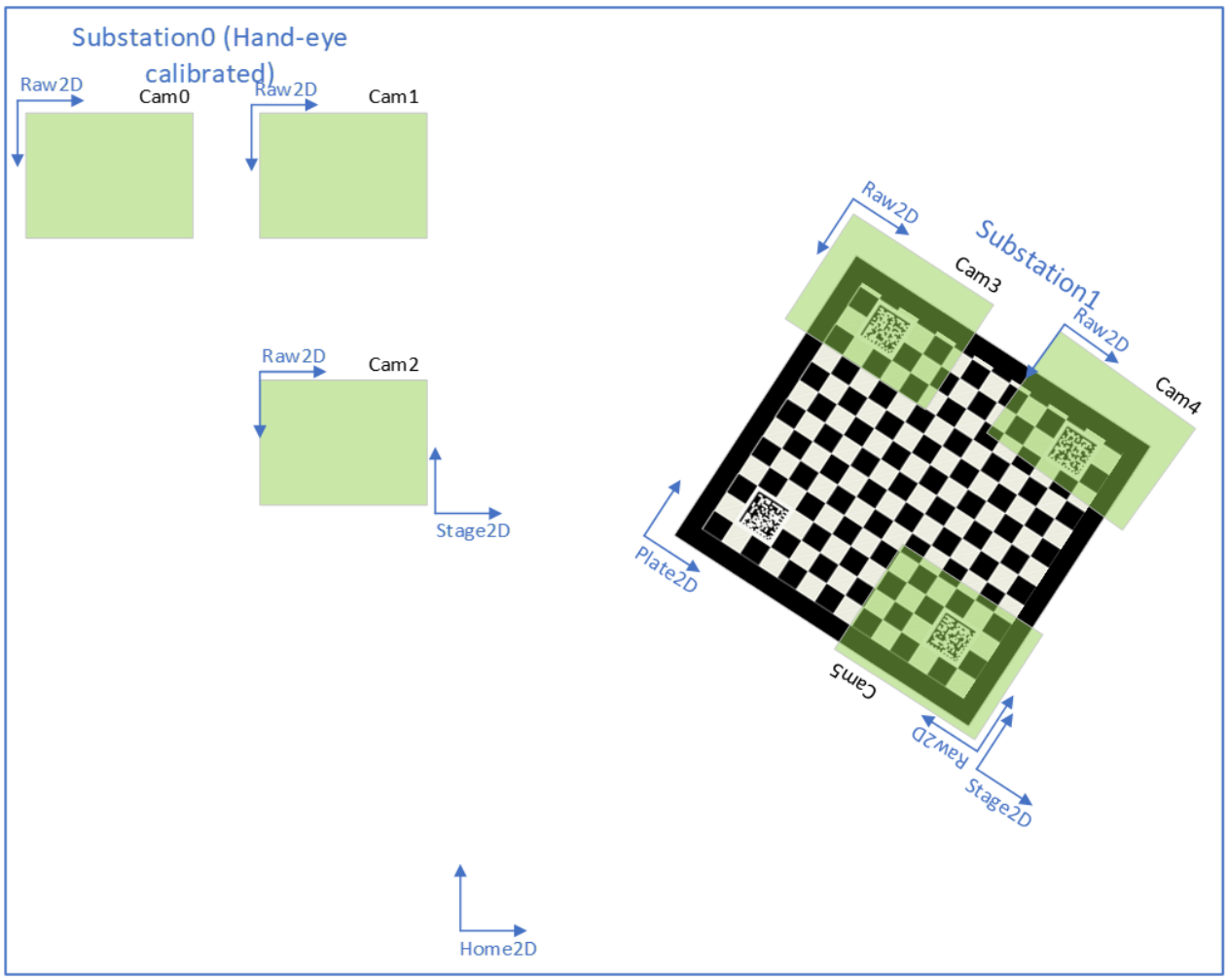
1. The gripper should present the part to the substation that has been hand-eye calibrated, SubStation0 in the figure above, and should send the calibration command along with gripper pose at this station.
2. The gripper should present the part to the substation that must be cross-calibrated, SubStation1 in the figure above, and should send the calibration command along with gripper pose at this station. Care should be taken to ensure that relationship between the part and the gripper does not change during the entire calibration process.

If feature finders needed to cross-station calibrate the cameras have not been added, they should be done. Feature finders in each station should be given unique names. Feature finders that find the same features across the stations (the hand-eye calibrated station and the station to be cross-station calibrated) should be given the same names in order to establish correspondence. If the feature finders have already been added, before the start of this step, the calibration process is complete. Otherwise Steps 1-4 should be repeated.

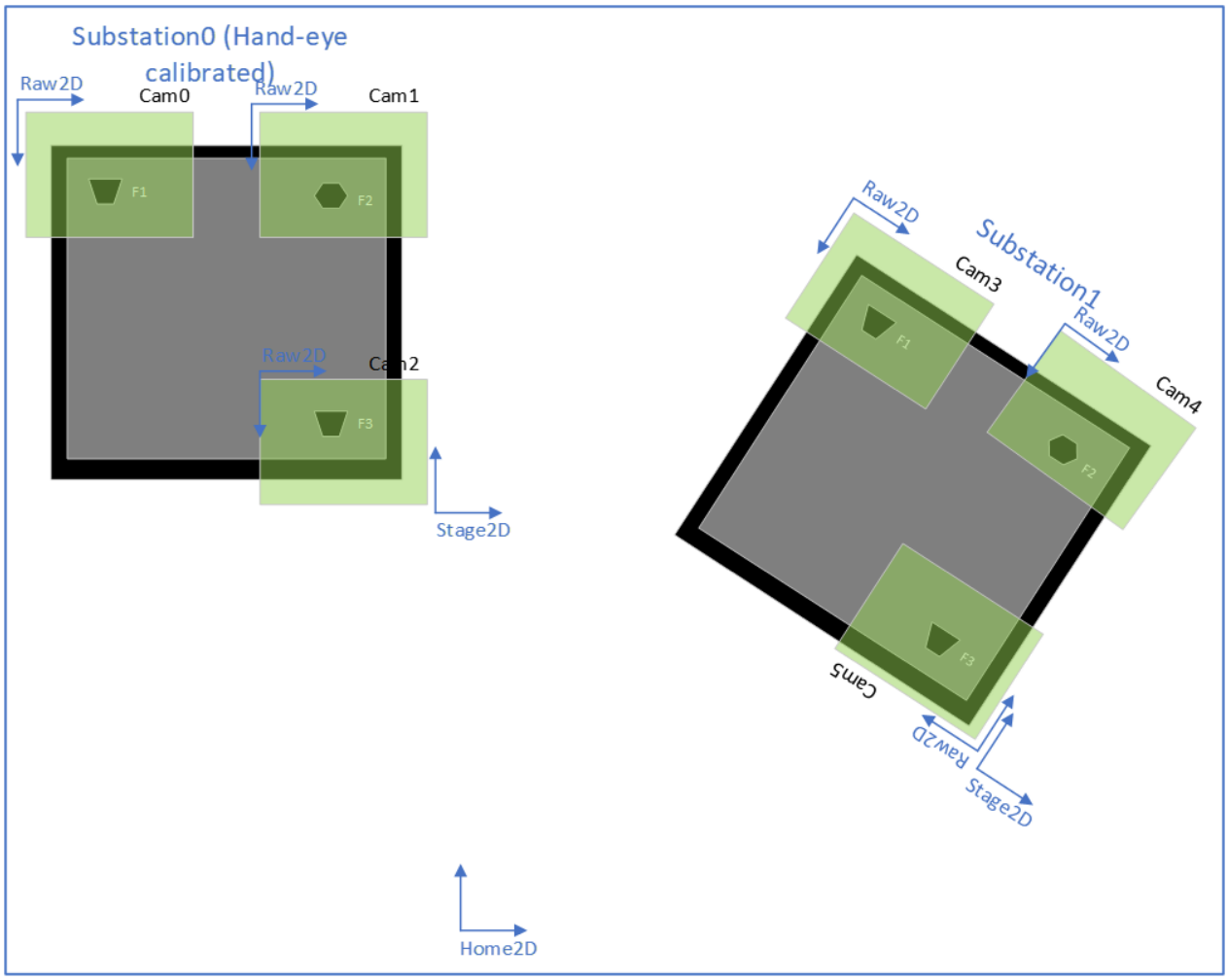
Hybrid Cross Calibration

When performing pure part based cross-station calibration, we assume that the lens does not introduce any non-linear distortion during imaging. The presence of lens distortion affects the accuracy of calibration adversely. The hybrid mode solves this problem. This mode of calibration uses two calibration targets. It first uses a checkerboard to train image correctors for all the cameras to be cross-station calibrated. This step models the nonlinear distortion in the imaging process. Then, it uses a part to cross-station calibrate the remote cameras.

The advantage of this technique is that during the second step of the calibration the features are found on a corrected image that is free of lens and perspective distortion.



The figure shows the cameras in SubStation1 being checkerboard calibrated. Following this calibration, all the cameras in the substation would be able to map the image features to Plate2D



The locations of the features F1, F2 and F3 are computed in Home2D in SubStation0. The corresponding feature locations are computed in SubStation1. Even though we need three features to perform cross-calibration, the three features need not be in the same camera because one can map the features in SubStation1 to Plate2D

Sequence of Commands

The following are the sequence of commands that must be sent from the PLC to AlignPlus to perform cross-station calibration for the above system

1. A checkerboard calibration plate should be placed in the substation to be cross-station calibrated and the calibration command with an arbitrary gripper must be sent. The pose of the gripper is not relevant in this step.
2. The gripper should present the part to the substation that has been hand-eye calibrated, SubStation0 in the figure above, and should send another calibration command along with the gripper position at this station. At this command, the vision system will acquire images from all cameras in SubStation0.
3. The gripper should present the part to the substation that must be cross-calibrated, SubStation1 in the figure above, and should send the third command along with the gripper position at this station. At this command, the vision system will acquire images from all cameras in SubStation1 and then process cross calibration. Care should be taken to ensure that relationship between the part and the gripper does not change during the entire calibration process.

If feature finders needed to cross-station calibrate the cameras have not been added, they should be done. Feature finders in each station should be given unique names. Feature finders that find the same features across the stations (the hand-eye calibrated station and the station to be cross-station calibrated) should be given the same names to establish correspondence. If the feature finders have already been added, before the start of this step, the calibration process is complete. Otherwise Steps 2 and 3 should be repeated.

Applications of Cross-Calibration

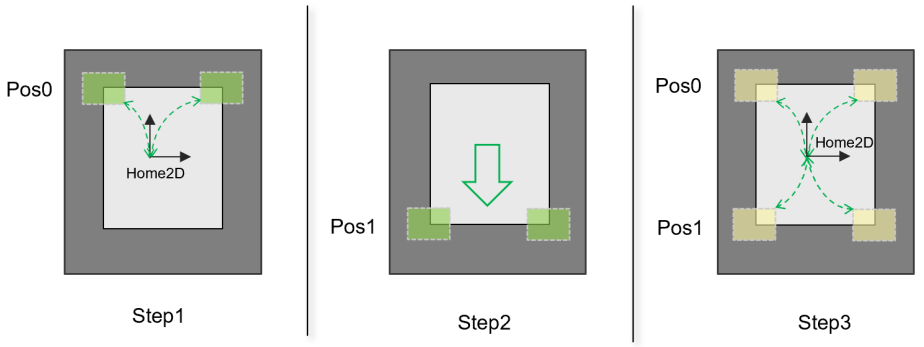
Shuttling Camera Calibration

A shuttling camera application is one where one or more cameras are mounted onto motion devices whose purpose is to reposition the cameras, for capturing images of a different regions of a part. The same result can be achieved by having dedicated cameras for each region to be imaged. Shuttling camera systems are used are to reduce the expense of additional vision hardware. An example of a shuttling camera applications is shown below. Here two cameras are shuttled between two positions, Pos0 and Pos1 to image all the corners of the part.

In practice it is possible to hand-eye calibrate the cameras at the four positions, either by capturing the images of the four corners at each pose of the motion device during hand-eye calibration process, or by performing two hand-eye calibrations, one at Pos0 and another at Pos1. However, the hand-eye calibration process takes more time to complete and might not be convenient.

To calibrate the system in our example the following steps can be executed

1. Perform hand-eye calibration at Pos0.
2. Capture an image of the calibration target at Pos0
3. With the calibration target being stationary, shuttle the cameras to Pos1 and capture the image of the target
4. Perform cross-calibration using the images from the previous two steps



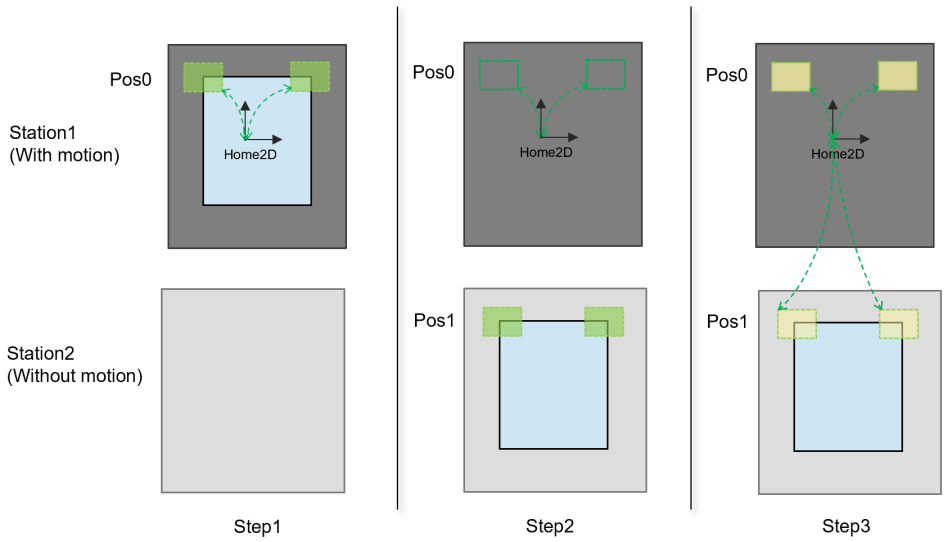
The calibration target could be calibration plate or calibration part with features. It can be manually placed in the beginning on stage/platform with vacuum on to keep it flat.

Lamination Systems

On some systems, it may not be possible to use a robot to assemble two parts due to mechanical or other constraints. These systems typically use a motion device, for example a UVW stage, that holds one of the parts to be assembled. A platform contains the second part. During run-time, the part on the motion device is manipulated so that it accounts for any variation in the positions of the two parts at their respective stations. After alignment, a gripper attached to a gantry, that simply shuttles between the two stations between two fixed positions, transfers one of the parts from its station to the other station to assemble the parts.

The figure below shows such a system. To calibrate the system the following steps can be executed

1. Perform hand-eye calibration at Station1.
2. Capture an image of the calibration target at Station1
3. With the calibration target being stationary, shuttle the calibration target to Station2 using the gripper that is used to assemble the parts and capture the image of the target
4. Perform cross-calibration using the images from the previous two steps.



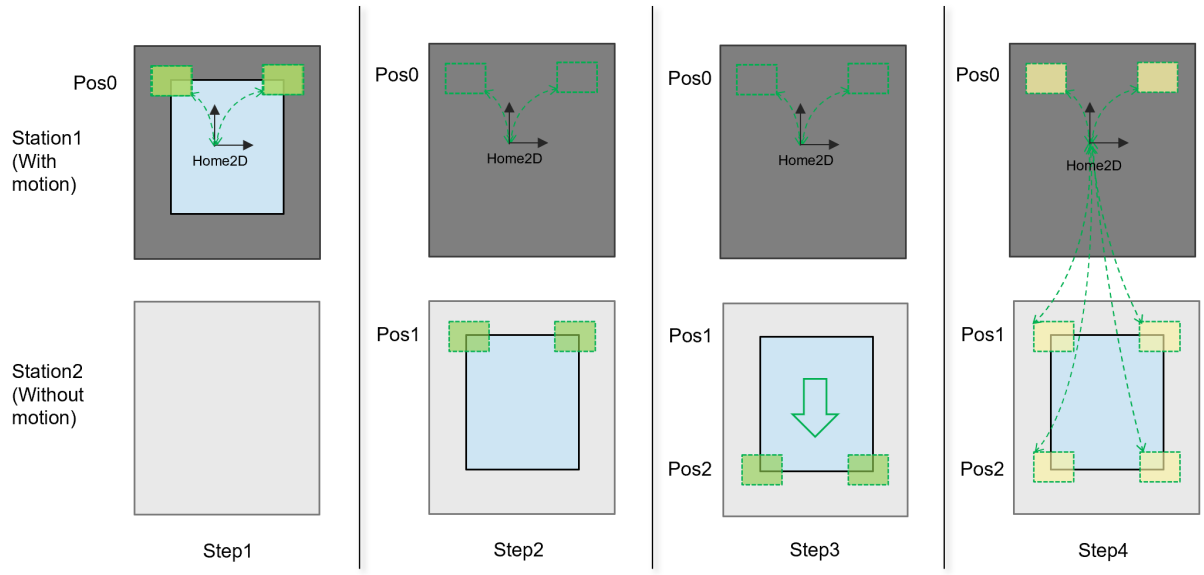
Note:
Here the station 1 and 2 can have different Z height which is perpendicular to Home2D plane.

The mechanical transfer of calibration target should be between the same positions as the run time part during assembly, using the same mechanical system. Failure to do so will result in invalid calibration data.

Another example

In the example above, cross calibration is conducted via the following steps:

1. The system is hand-eye calibrated.
2. Images of the calibration target using the cameras that have been hand-eye calibrated are taken.
3. The calibration target is transferred from stage to platform. The images of the calibration target are captured at Pos1
4. The cameras are shuttled to Pos2 using a different mechanical system, and images of the calibration target are acquired at Pos2.
5. Cross calibration results are computed using the images captured in the above three steps. This result will enable mapping of image features in images at Pos0, Pos1 and Pos2 to Home2D



Checkerboard Calibration

Checkerboard Calibration uses calibration plate to compute intrinsic parameters such as lens distortion and perspective distortion and generate images that are distortion free. After checkerboard calibration, a transform between Raw2D and Plate2D will be established. Checkerboard calibration targets are also used in other calibrations such as hybrid hand-eye calibration, hybrid cross calibration or hybrid manual calibration. In all these calibrations, the checkerboard is used to compute intrinsic parameters. The other calibration parameters are computed on the images that are free of these non-linear distortions.

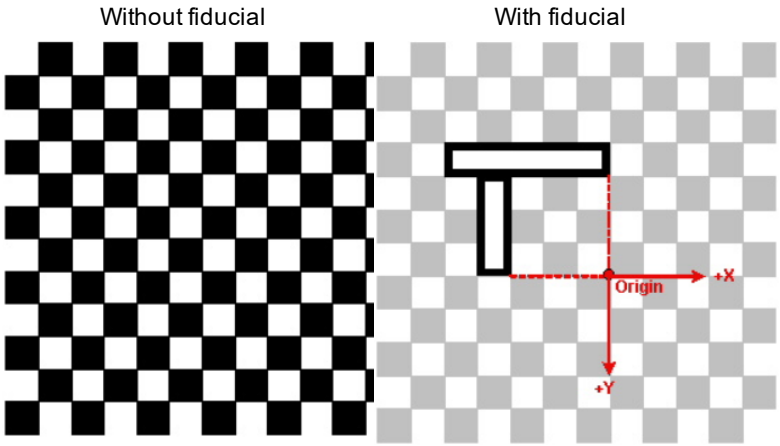
Calibration plate

There are two major type of checkerboard plate used in AlignPlus.

- **Standard Calibration Plate**

  Standard calibration plate is a plate with intersective white and black tiles, every tile shares the same height and width. Typical pitch(the distance between successive corresponding points or lines) of different calibration plates are 0.5mm, 1mm, 2mm, 5mm and 10mm. The tiles provide references for vision on how real equal-distance points are distributed under the camera, thus help vision to calculate how much lens or perspective distortions the camera has.

  As for Plate2D definition, if the plate doesn't have any fiducial indicating x, y directions, then user needs to manually selecting x,y axes and origin on calibration image for Plate2D. However if a "L" shape fiducial is seen in FOV, then vision will automatically define Plate2D based on the L-shape fiducial(shown as below).



- **DataMatrix Calibration Plate**

  DataMatrix Calibration plate uses DataMatrix codes to label the locations of multiple grid vertices on the plate with unique Plate2D origin (each DataMatrix code has different coordinates).

  Some DataMatrix code not only has coordinates information, but also indicates plate's grid pitch. Based on this, DataMatrix calibration plate are classified to two types:

  - DataMatrix without Grid Pitch

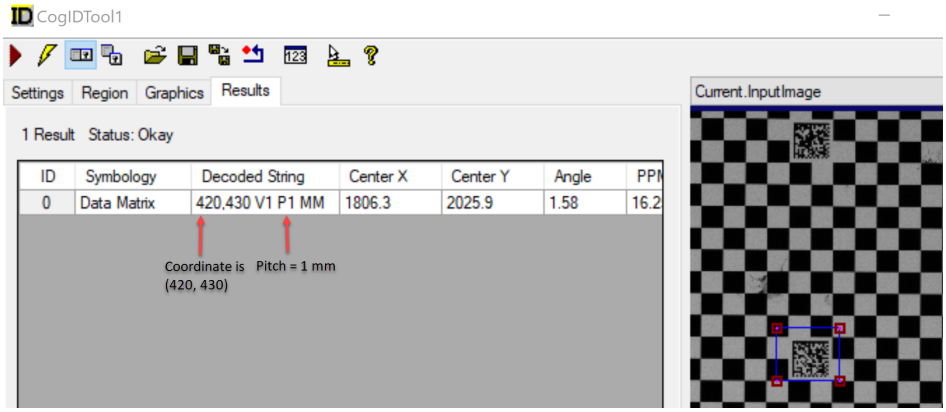
    User need to input pitch on UI manually

  - DataMatrix with Grid Pitch

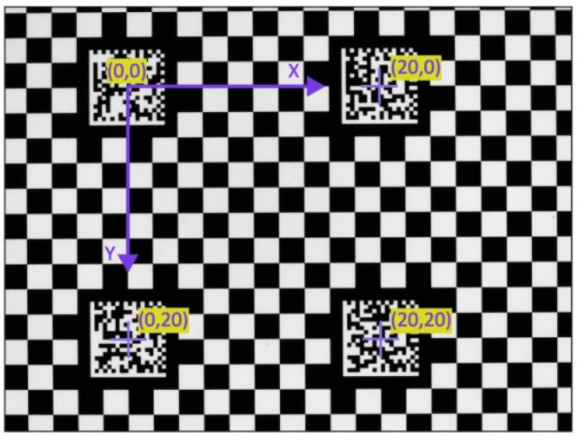
    User doesn't need to input pitch, vision will automatically identify the pitch size by reading information from DataMatrix code.

  However, the two types have no difference in appearance to human eyes, the only difference is DataMatrix code content. To tell which type it is, user can run CogIDTool in VisionPro ToolBlock to get the content of it.

  Here is an example of DataMatrix with Grid Pitch:



  The following figure shows how multiple DataMatrix fiducial marks serve to label four vertices on a checkerboard calibration plate (in this case, a plate with a 2mm grid pitch)



For information about checkerboard vertex extraction and Plate2D definition, please refer to Feature Extractor on page 57.

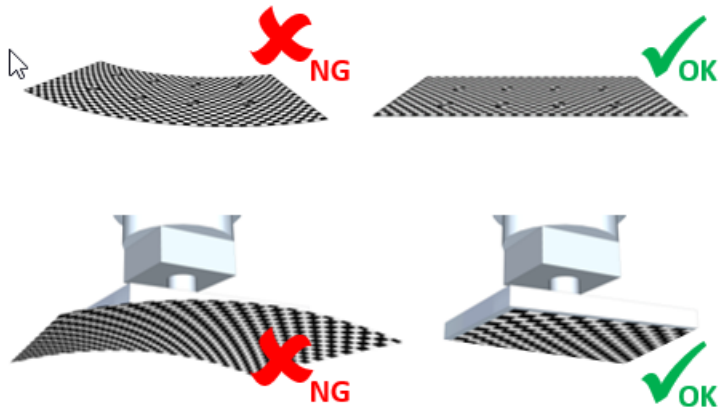
Train Time

Checkerboard plate should be rigidly attached to an precision motion stage before hand. The checkerboard plate material could vary from stainless steel to pottery to transparent mylar sheet depending on applications. Mylar sheet is widely used as it's thin enough to be picked up by robot or gripper using vacuum, and it has two identical and transparent sides which can be easily transferred between upward surface and downward surface and captured by down-looking camera or up-looking camera.

Once the plate is in position, take a picture of the calibration plate, and run checkerboard calibration to get the calibration result.

For calibration best practice, it is recommended:

1. Calibration plate cover the whole FOV
2. Keep calibration plate flat



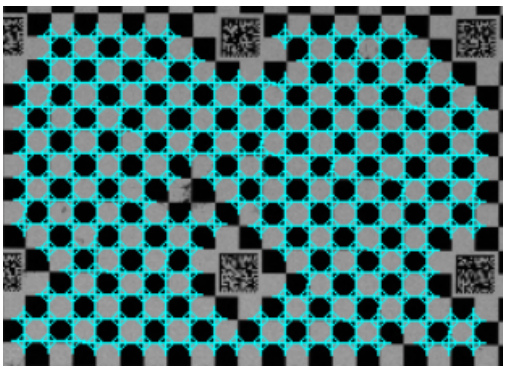
3. Make sure calibration plate is within focus
4. Keep plate clean so the coverts are extracted correctly
5. Adjust lighting or exposure time to make white and black tiles have good contrast without oversaturation

Calibration Result Check

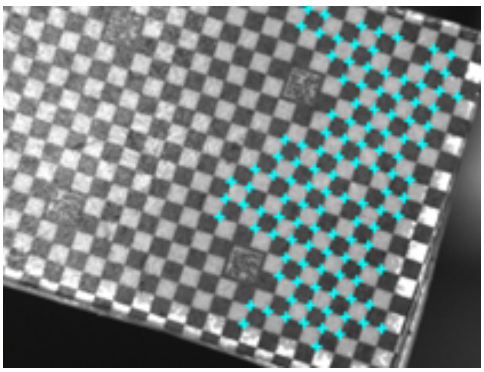
- Feature extraction rate

  It is recommended that over 70% of the checkers within FOV should be extracted. The calibration is accurate in areas with extracted checkers. In areas without checker features, the calibration is computed by extrapolation, which will result in low accuracy.

  High extraction rate

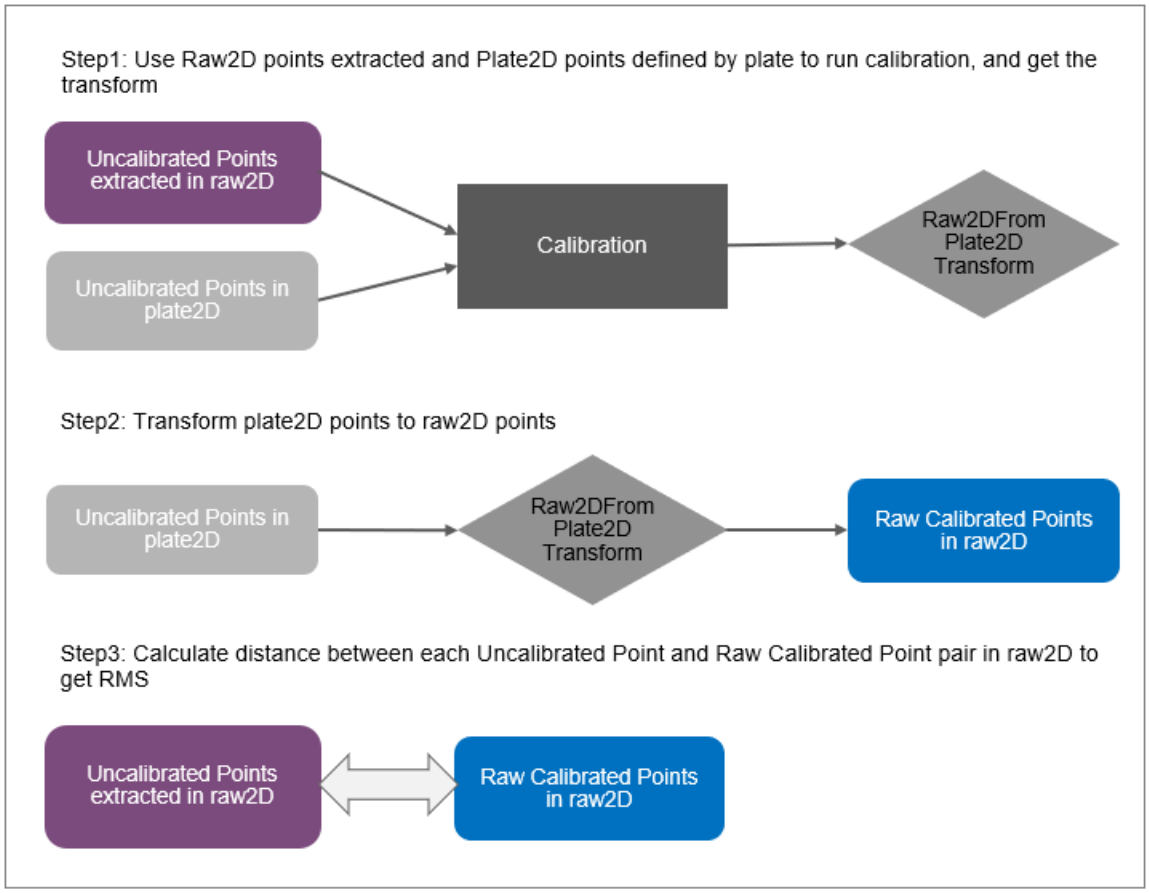


Low extraction rate

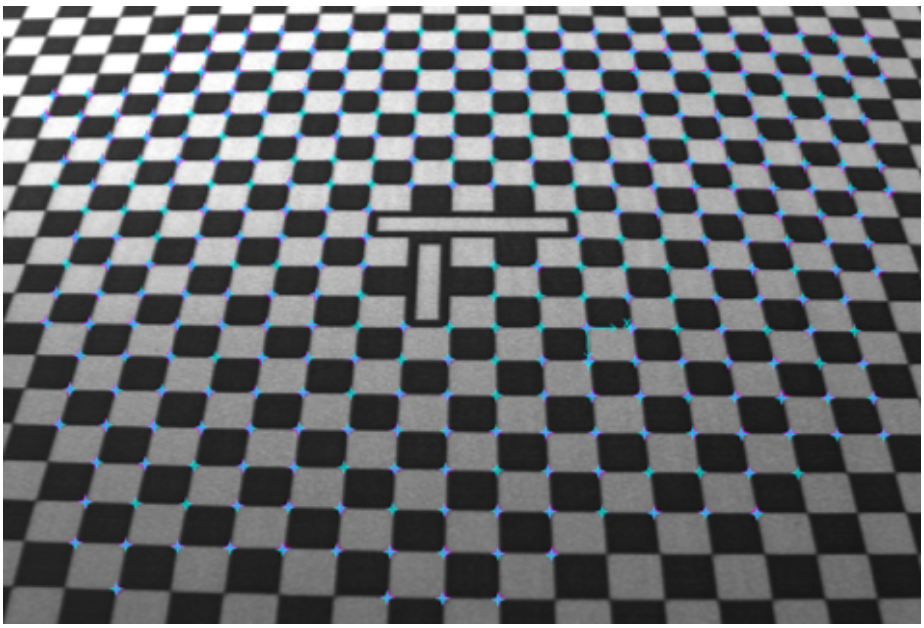


- RMS

RMS(Root Mean Square) indicates checkerboard calibration result by comparing differences between uncalibrated points and raw calibrated points.



The diagram above shows how RMS error is computed during checkerboard calibration.



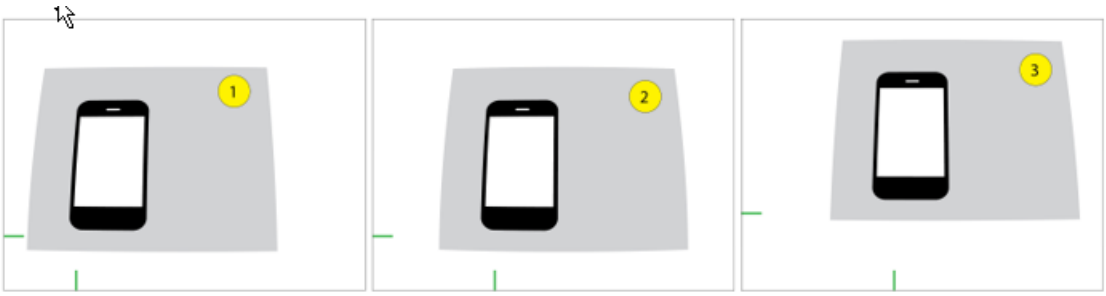
$$RMS = \sqrt{\frac{\sum_{i=1}^{n} e_i}{n}}$$

For high-accurate application, RMS error in Raw2D should be within 1 pixel.

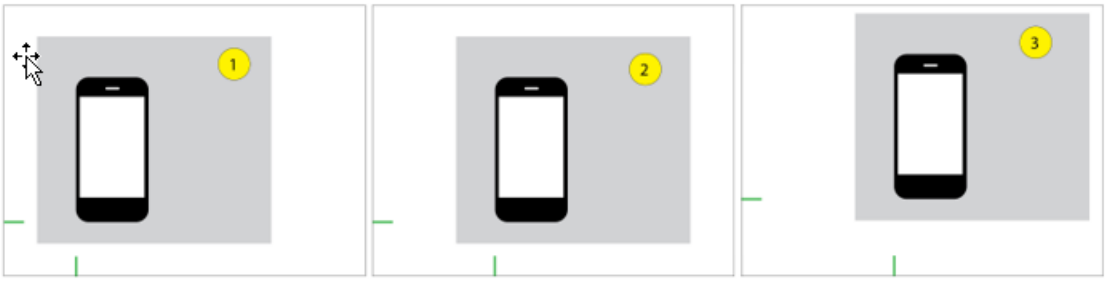
Run Time

Checkerboard calibration computes intrinsic parameters such as lens and perspective distortion. They also generate image correctors that take the image captured by the camera as an input and generate images that are free of these distortions. Many of Cognex's vision tools assume that the relationship between Raw2D and Client2D is an affine transform. The correctors generate images for which this assumption holds true.

- raw images



- corrected images



User can also choose specific lens distortion mode for image correction depending on the lens type used in application, please refer to Lens Distortion Mode on page 56.

Lens Distortion Mode

Select one of the following calibration Calibration Mode values.

- **NoDistortionWarp**

  This mode will model perspective distortion only; any nonlinear optical distortion is ignored. By comparing the residual error values produced using this computation mode with the residual error values from ThreeParamRadialWarp or SineTanLawProjectionWarp you can improve your understanding of the individual sources of residual error.

- **ThreeParamRadialWarp**

  This model calibrates for nonlinear optical distortion and perspective distortion. When compared with PerspectiveAndRadialWarp, this mode adds additional coefficients that properly model the location of the optical center.

  Note: This mode is recommended for lenses with minimal to moderate distortion, typically those with focal lengths greater than 6mm.

- **SineTanLaw**

  This model calibrates for nonlinear optical distortion and perspective distortion. When compared with ThreeParamRadialWarp, this model uses a computation model that is appropriate for lenses with moderate to severe distortion, typically those with focal lengths less than 6mm.

Feature Extractor

Algorithm

This value specifies how the tool will search the calibration image for calibration vertex points. This method must match the type of calibration plate that you are using.

- **Standard**

  VisionPro supports this mode for existing applications but does not recommend this for new applications. For checkerboard plates with Cognex 'L' fiducial marks or without a fiducial mark, use Exhaustive or Efficient. For checkerboard plates with DataMatrix fiducials, use Exhaustive Multi Region or Efficient Multi Region.

- **Exhaustive**

  This mode provides the most robust feature extraction possible for checkerboard plates (with or without the Cognex "L" fiducial marks) but requires the longest time to execute. Choose the Efficient mode for generally faster processing times.

- **Exhaustive Multi Region**

  This mode provides the most robust feature extraction possible for checkerboard calibration plates with DataMatrix fiducial marks but requires the longest time to execute. Choose the Efficient Multi Region mode for generally faster processing times. This method can tolerate partial occlusion of plate features from reflections or other issues, as long as the fiducial marks are visible.

- **Efficient**

  This mode provides generally good calibration results for checkerboard plates (with or without the Cognex "L" fiducial marks) in less time than Exhaustive mode, provided certain conditions apply.

- **Efficient Multi Region**

  This mode provides generally good calibration results for checkerboard plates with DataMatrix fiducial marks in less time than CheckerboardExhaustiveMultiRegion mode, provided certain conditions apply.

Label Mode

This value specifies the type and characteristics of the fiducial marks on the calibration plate.

Note:
Not all combinations of algorithm and label mode are valid. If you enter an invalid combination, the tool will generate an error at calibration time.

- **None**

  Use this value for calibration plates with no fiducial marks. The Calibration tool will use the tile vertex or dot center that is closest to the center of the image as the origin and will assign the positive x-axis and y-axis to the grid lines which are closest in angle to the x-axis and y-axis in the pixel coordinate space of the calibration image.

- **UseOrigin**

  This specifies that the vertex closest to a designated origin location be used as the origin of the Plate2D coordinate system. The designated origin can be set using the properties Origin X and Origin Y. The direction of the axes is set exactly as in None label mode.

- **UseFiducial**

  This specifies that the plate uses a single standard Cognex 'L' fiducial mark to be located and used for setting the Plate2D coordinate system.

- **UseDataMatrix**

  This specifies that standard Data Matrix codes are to be used as calibration fiducials that are uniquely identifiable in the calibration target. This mode will ignore any information about the physical grid pitch that may be encoded in the Data Matrix codes, and will instead use Physical Grid Pitch X and Y properties.

- **UseDataMatrixWithGridPitch**

  This specifies that the plate includes Cognex-compatible DataMatrix fiducial marks, each of which specifies the coordinates of the mark in the plate's 2D coordinate system and the grid pitch in mm.

  If you specify this mode, the tool ignores any grid pitch values that you supply.

Manual Calibration

When it is not feasible to perform hand-eye or cross-station calibration, manual calibration can be used to establish the relationship between Raw2D and Home2D. The Raw2DFromHome2D transform is an affine transform. In the manual calibration mode, a suitable HMI allows the user to calibrate using various techniques. The users will have the option to either 1. Enter the affine parameters directly, 2. Identify features in the field of view whose position is known in Home2D, 3. A combination of the above two. Those inputs could be translation, rotation, scale, aspect, and skew of the transform or paired points in Raw2D and Home2D depending on which method user choose.

Manual Calibration Modes

AlignPlus supports two major modes of manual calibration.

- Non-hybrid mode

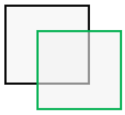
  In this mode, the calibration process computes an affine transform that can be used to map features from uncalibrated space to calibrated space. This mode cannot be used to model any lens and perspective distortion

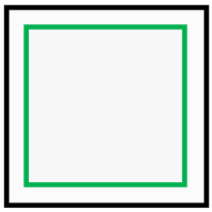
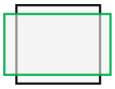
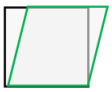
- Hybrid mode

  In this mode, the lens and perspective distortion are removed through checkerboard calibration. The HMI allows the user to compute a 2D rigid transform and flip in handedness, that maps uncalibrated features to calibrated features.

Affine Transform Parameters

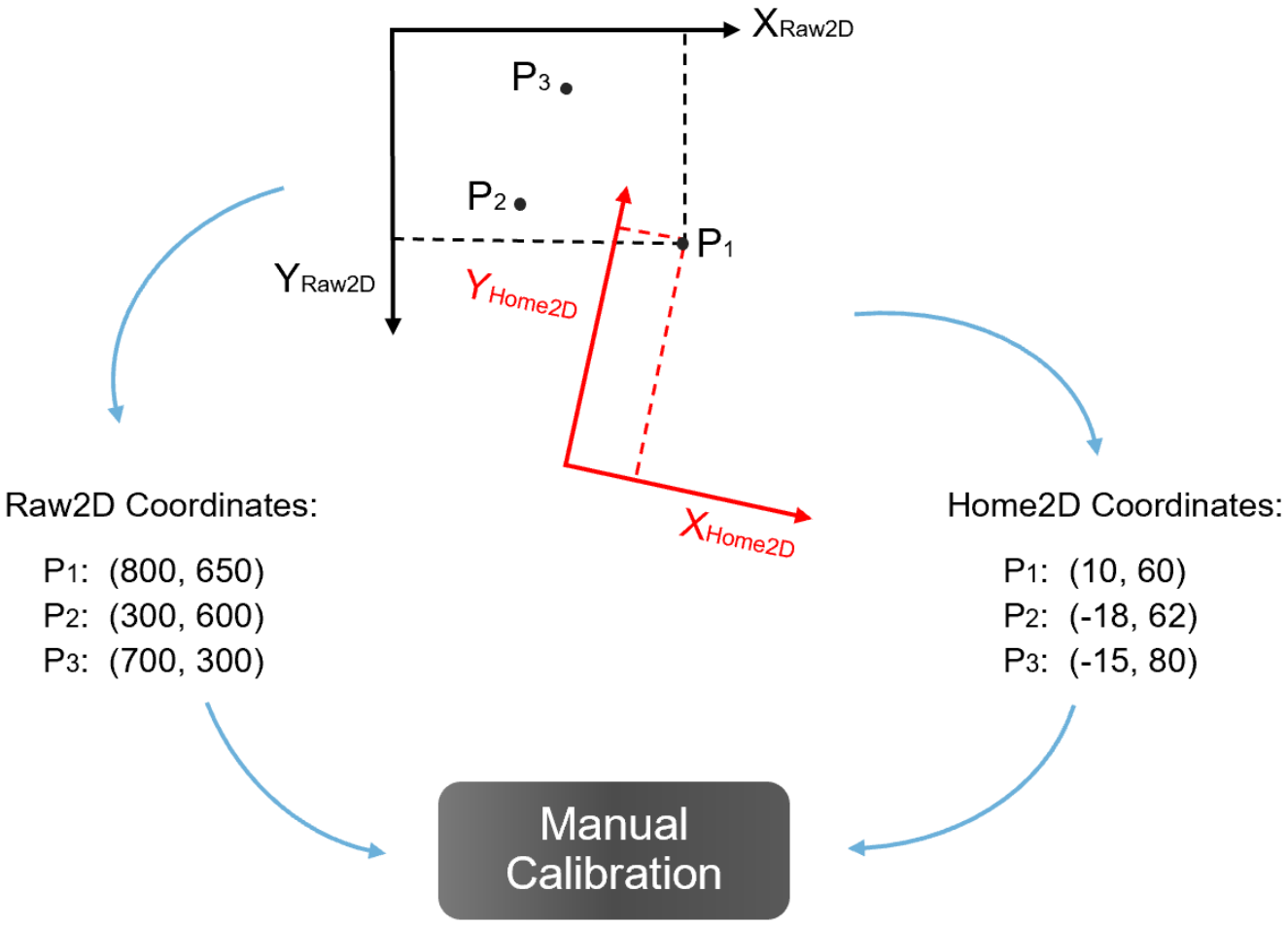
This method allows user directly input affine transform parameters include translation, rotation, scale, aspect, and skew. Those parameters can be obtained through mechanical measurement after cameras are installed on machine.

Parameters	Description	Example
Translation	Specifies an offset of a Camera’s center relative to Home2D space	
Rotation	Specifies rotation of camera relative to Home2D space	

Scaling	Specifies the x- and y-axis units of one Raw2D relative to Home2D. This is the Pixel Size.	
Aspect	Specifies the ratio of scaling along the y-axis over scaling along the x-axis in Raw2D. When x and y axes share the same scaling, aspect is 1.	
Skew	Specifies x/y axis angle compared with original angle in Raw2D, when x and y axes are perpendicular to each other, skew is 1.	

Paired Points

The Manual Calibration HMI also allows computation of Raw2DFromHome2D by specifying feature locations in Raw2D and Home2D. AlignPlus also provides HMIs that allow the user to specify a subset of the affine parameters and compute the rest of the parameters using point correspondence.



Transforms in Hand-eye Calibration Result

Besides getting transforms from Space Tree, the following transforms are available in hand-eye calibration result.

General Transforms

These transforms are valid both in stationary and moving camera configurations.

Home2DFromStage2D

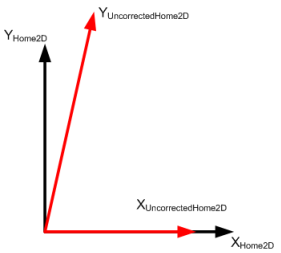
This is the actual physical pose of the motion stage in Home2D. It defines the relationship between Home2D and Stage2D and is represented by a 2D rigid transform.

In the stationary camera configuration, this is the motion of the motion stage of the object. In the moving camera configuration, this is the motion of the motion stage onto which the cameras are attached – the object platform is stationary in this case.

UncorrectedHome2DFromStage2D

This is the commanded pose of the motion stage.

Due to systematic errors in the motion stage, the commanded pose of the motion stage may not match the actual physical pose of the motion stage in Home2D (that is, Home2DFromStage2D). UncorrectedHome2DFromStage2D is used to describe the commanded pose of the motion stage because it represents your best guess of the pose of the motion stage prior to calibration. UncorrectedHome2D has its X and Y axes along the X and Y axes of stage, the origin of UncorrectedHome2D is the same as Home2D.



Raw2DFromCamera2D

The transform that maps coordinates in the camera coordinate system (Camera2D) to the image coordinate system (Raw2D). This is independent of the motion stage pose.

Raw2DFromHome2D

The mapping between Raw2D and Home2D. In the moving camera configuration, this transform is dependent on the actual UncorrectedHome2DFromStage2D pose.

Stationary Camera Configuration transforms

These transforms are valid in the stationary camera configuration only.

Home2DFromStationaryCamera2D

This is the placement pose of the camera, that is, this specifies how the camera was placed in the system. The placement pose of the cameras is necessary for mapping a physical position in camera coordinates to Home2D. This is a rigid transform that may also flip handedness.

Stage2DFromMovingPlate2D

This is the placement pose of the calibration plate, that is, this specifies how the calibration plate was placed in the system. The placement pose of the calibration target is necessary for mapping a physical position on the calibration target to Home2D. This is a rigid transform that may also flip handedness.

Moving Camera Configuration transforms

These transforms are valid in the moving camera configuration only.

Home2DFromStationaryPlate2D

This is the placement pose of the calibration plate, that is, this specifies how the calibration plate was placed in the system. The placement pose of the calibration target is necessary for mapping a physical position on the calibration target to Home2D. This is a rigid transform that may also flip handedness.

Stage2DFromMovingCamera2D

This is the placement pose of the camera, that is, this specifies how the camera was placed in the system. The placement pose of the cameras is necessary for mapping a physical position in camera coordinates to Home2D. This is a rigid transform that may also flip handedness.

Feature Finding

Feature finding is a process of using vision tools to locate certain features on part. During alignment/assembly of parts, AlignPlus locates features in a shared coordinate space, Home2D. The feature locations are used by the pose computation task to guide the motion device so that the desired alignment/assembly characteristics are obtained.

This chapter addresses the types of features that AlignPlus supports and the types of finders that are available for finding these features.

Features

Features are characteristics that define a part. A set of features on a part can be used to define the pose. Some examples of features are, fiducials on a part, edges of the part, and corners of the part. AlignPlus allows location of three types of features on the part.

- Point Feature

  Point features are defined by their rotation and position in the image. Some examples of point features are fiducials, corners, centers of blobs etc. Cognex provides several feature finders to find these types of features. Some of them are CogPMAlignTool, CogBlobTool, CogCornerFinderTool etc

- Line Feature

  Line features include lines and line segments. It can be found by CogFindLineTool, CogLineMaxTool or CogSmartLineMaxTool.

- Generic Feature

  Generic features are features of type System.Object. Since all .NET objects derive from System.Object, these features can be of any .NET type. This provides the user the flexibility to locate any type of feature such as lines, angles, fiducials etc. The user would be responsible to write a pose computer that can consume these feature locations. Besides being used for alignment/assembly, these features can be used for inspection of parts.

Features Finder

AlignPlus uses various types of features finders to locate the various types of features. Point, line and generic features finders are used to locate point, line and generic features respectively. Each features finders receive images from the multiple cameras that capture the image of a part, locate the features using the finders that are added by the user, and output a list of found features. The user can use HMIs that are available in the application to add the finders required to locate the features. The HMIs are designed to ease the addition and setup of the finders.

Feature Finder

A feature is a characteristic of a part. It can be a point feature such as corner or fiducial, a line feature or generic feature. A finder is a tool that is used to locate this feature. AlignPlus allows one to add multiple finders that locate the same point feature. By default, when a finder is added, it is associated with a unique feature with the same name as the finder. The user has the ability to reassign the feature associated with the finder. AlignPlus uses the location computed by the finder with the highest score during pose computation.

For all line and generic features, the user can associate one feature with one finder.

Here is a summary of all available **Features Finders** supported by AlignPlus:

Features Finder Type	Feature Finder Type	VisionPro Tools Beneath	Function
Point Features Finder	PatMax Pattern Finder	CogPMAlignTool	Find a pattern on part
	Corner Finder	CogFindCornerTool	Find a corner point
	Custom Tool Block Point Finder	CogToolBlock + other tools user choose	Customized VisionPro Toolblock to find a point
Line Features Finder	Line Finder	CogFindLineTool	Find line segment on part
	Custom Tool Block Line Finder	CogToolBlock + other tools user choose	Customized VisionPro Toolblock to find a line segment
Generic Features Finder	Custom Tool Block Generic Finder	CogToolBlock + other tools user choose	Customized VisionPro Toolblock to find a generic feature that is defined by user

The output features are stored in the following three type of AlignPlus classes:

- CogAlpsPointFeature

  Stores x, y and theta of found point feature using a point features finder.

- CogAlpsLineFeature

  Stores line segment of found line feature using a line features finder.

- CogAlpsGenericFeature

  Stores user defined feature located using a generic features finder.

Feature Class

CogAlpsPointFeature

Class that represents a point feature.

Class name: CogAlpsPointFeature

Namespace: Cognex.Designer.AlignPlus.Alignment

Assembly: Cognex.Designer.AlignPlus.Alignment.dll

Properties

Name	Type	Description
CameraIndex	Int32	The index of the camera containing the point feature finder and the feature location. The value is -1 for multi camera custom feature finder.
Feature	Object	Not used.
FeatureLocationX	Double	The X coordinate of the found feature. This value is undefined if IsFound = false.
FeatureLocationY	Double	The Y coordinate of the found feature. This value is undefined if IsFound = false.
FeatureRotationXIn Degrees	Double	The angle of the X Axis of the found feature. This value is undefined if IsFound = false and is valid only if found by a PatMax tool.

Name	Type	Description
FeatureRotationYIn Degrees	Double	The angle of the Y Axis of the found feature. This value is undefined if IsFound = false and is valid only if found by a PatMax tool.
FeatureTransform	Object	Not used.
FeatureName	String	Indicates the name of the feature.
FinderName	String	The name of the VisionPro tool.
IsFound	Boolean	Indicates if the feature was found.
IsValid	Boolean	Indicates if the results of the finder is valid. If this is true then the X, Y and Score properties are valid. If MultipleFindersPerFeatureMode is true, IsValid can be true, but IsFound can be false.
Score	Double	The score for the PatMax tool. For other tools the value is -1
UserData	Cognex.VisionPro.CogDictionary	Any user data that is generated by the finder.

CogAlpsLineFeature

Class that represents a line feature.

Class name: CogAlpsLineFeature

Namespace: Cognex.Designer.AlignPlus.Alignment

Assembly: Cognex.Designer.AlignPlus.Alignment.dll

Properties:

Name	Type	Description
CameraIndex	Int32	The index of the camera containing the point feature finder and the feature location. The value is -1 for multi camera custom feature finder.
Feature	object	The found feature, here it is referring to LineSegment
FinderName	String	The name of the VisionPro tool.
IsFound	Boolean	Indicates if the feature was found
LineSegment	Cognex.VisionPro.CogLineSegment	The found line segment. This value is undefined if IsFound = false
UserData	Cognex.VisionPro.CogDictionary	Any user data that is generated by the finder.

CogAlpsGenericFeature

Class that represents a generic feature.

Class name: CogAlpsGenericFeature

Namespace: Cognex.Designer.AlignPlus.Alignment

Assembly: Cognex.Designer.AlignPlus.Alignment.dll

Properties:

Name	Type	Description
CameraIndex	Int32	The index of the camera containing the point feature finder and the feature location. The value is -1 for multi camera custom feature finder.
Feature	Object	The found feature. This value is undefined if IsFound = false
FinderName	String	The name of the VisionPro tool.
IsFound	Boolean	Indicates if the feature was found
UserData	Cognex.VisionPro.CogDictionary	Any user data that is generated by the finder.

Pose Compute

Pose Compute Overview

Alignment or assembly systems use motion devices to move one or more parts so that a part with desired alignment/assembly characteristics is obtained. The pose computation task computes the motion device parameters that facilitates this goal.

Alignment Applications

The goal of alignment applications is to align a single part to a desired pose. These types of applications use the feature locations on a part to be aligned and compare them to the corresponding feature locations on a part with the desired pose, established during a training step, to compute the motion device parameters. AlignPlus supports two types of alignment applications based on the technique to handle the parts.

Part Handling Technique

Alignment applications support two types of part handling techniques:

- Align To Base

  In these types of alignment systems, the motion device is in engagement with the part. During alignment, the application computes motion device parameters which move the run-time part so that it has the same pose as the train-time part, in Home2D. See more about Align to Base on page 8

- Align To Gripper

  In these types of alignment systems, the run-time part rests on a platform. During alignment, the application computes motion device parameters that move the motion device so that the relationship of the motion device to the run-time part on the platform is the same as one established during a training step. In these types of applications, the train-time and run-time parts have the same pose in Stage2D (when a robot is used as a motion device, Stage2D is the coordinate system established by the gripper). See more about Align to Gripper on page 9.

Application	Part Handling Type	Feature Compare Mode	Train Time	Run Time		
				Part pose difference (Calculated based on features)	Motion Move	Motion Move Starting Pose
Alignment	Align To Base	Golden Pose	i. Features golden pose	• Run time -> Golden pose	Run time -> Golden pose	Stage's current pose
	Align To Gripper	Golden Pose	i. Feature's golden pose ii. Gripper's picking pose		Golden pose -> Run time	Arbitrary pose depending on user

Assembly Applications

Assembly applications align one part to another part so that the assembled part has the desired assembly characteristics. Assembly applications can be classified in various ways.

Part Handling Technique

- Assembly Blind Transfer

  These systems have one of the parts(Part A) to be assembled at a station on a motion device. The second part(Part B) is in another station, on a platform. The two parts are assembled by moving the parts from the first station to the second station using a shuttling device that moves repeatably between two fixed positions. Before transfer, the part on the motion device(Part A) is re-positioned such that when assembled, the assembled part has the desired assembly characteristics. This technique is also used in applications where a robot grips a part (Part A) at the first station and assembles this part to another part (PartB) that is on a platform at a second station. See more about Assembly Blind Transfer on page 14

- Assembly Guided Pick

  These types of systems use a robot to assemble parts. The motion parameters allow the robot to pick the part(Part A) at the pick-station correctly. After the part is picked, the part is placed on a part at the place-station by moving the robot to a fixed position that is established during training. See more about Assembly Guided Pick on page 17.

- Assembly Guided Place

  These types of systems use a robot to assemble parts. The motion parameters allow the robot to pick the part(Part A) at the pick-station by moving the robot to a fixed position that is established during training. After the part is picked, the part is placed on a part at the place-station correctly. The part is placed in such a fashion that following assembly the assembled part has the desired assembly characteristics. See more about Assembly Guided Place on page 18.

Part Pose Change Computation Technique

- Using Paired Features

  These types of assembly systems assemble parts by matching features of the pick-part to the corresponding features on the place-part and assemble the two parts such that the distance between the corresponding features minimize a metric in a least square sense. These systems employ a training step where the robot sends the gripper pose on one.

  During run-time, the alignment system performs the following steps:

  1. It locates the features on the part(Part A) at the pick station and the part(Part B) at the place station, both in Home2D.
  2. It computes a TrainFromRun transform that would map Part A's run-time features to Part B's run-time features.
  3. It uses the above transforms along with the train time pick-pose or train time place-poses of the grippers and to compute alignment parameters that would assemble the parts correctly.

- Using Golden Pose

  These types of systems assemble parts by computing the changes in the run-time pose of parts from their train-time poses in their respective stations. The recommended training steps for these systems are as follows:

  1. A correctly assembled part is placed at the place station.
  2. A robot picks the part that lives on the pick station(Part A) from the assembled part. The robot sends its pose during this step to the alignment system. The pose will be referred to as the train time place-pose of the gripper.
  3. The robot transfers the part(Part A) and places it at the pick station. The robot sends its pose following this step to the alignment system. The pose will be referred to as the train time pick-pose of the gripper.
  4. The alignment system locates the features on the part at pick station(Part A) in Home2D. The features will be referred to as the train time pick features.
  5. The alignment system locates the features on the other part at the place station(Part B) in Home2D. The features will be referred to as the train time place features.

  During run-time, the alignment system performs the following steps:

  1. It locates the features on the part at the pick station(Part A) in Home2D and computes the TrainFromRun transform for Part A at the train station. The TrainFromRun transform is one that would map Part A's run-time features to its train-time features.
  2. It locates the features on the part at the place station(Part B) in Home2D. It computes the TrainFromRun transform for Part B at the place station. The TrainFromRun transform is one that would map Part B's run-time features to its train-time features.
  3. It uses the above transforms along with the train time pick-pose and train time place-pose of the grippers and to compute alignment parameters that would assemble the parts correctly.

<table>
<tr><th rowspan="2">Application</th><th rowspan="2">Part Handling Type</th><th rowspan="2">Feature Compare Mode</th><th rowspan="2">Train Time</th><th colspan="3">Run Time</th></tr>
<tr><th>Part pose difference (Calculated based on features)</th><th>Motion Move</th><th>Motion Move Starting Pose</th></tr>
<tr><td rowspan="5">Assembly</td><td rowspan="2">Assembly Blind Transfer</td><td>Paired Features</td><td>/</td><td>• Part A Run time -> Part B Run time</td><td rowspan="2">Part A Run time -> Part B Run time</td><td rowspan="2">Stage's current pose</td></tr>
<tr><td>Golden Pose</td><td>i. Part A: features golden pose
ii. Part B: features golden pose</td><td>• Part A: Run time -> Golden pose
• Part B: Run time -> Golden Pose</td></tr>
<tr><td rowspan="2">Assembly Guided Pick</td><td>Paired Features</td><td>i. Gripper's place pose</td><td>• Part A Run time -> Part B Run time</td><td rowspan="2">Part B Run time -> Part A Run time</td><td>Arbitrary pose depending on user</td></tr>
<tr><td>Golden Pose</td><td>i. Part A: features golden pose
ii. Part B: features golden pose
iii. Gripper's pick pose and place pose</td><td>• Part A: Run time -> Golden pose
• Part B: Run time -> Golden Pose</td><td>Arbitrary pose depending on user</td></tr>
<tr><td rowspan="2">Assembly Guided Place</td><td>Paired Features</td><td>i. Gripper's pick pose</td><td>• Part A Run time -> Part B Run time</td><td rowspan="2">Part A Run time -> Part B Run time</td><td>Arbitrary pose depending on user</td></tr>
<tr><td>Golden Pose</td><td>i. Part A: features golden pose
ii. Part B: features golden pose
iii. Gripper's pick pose and place pose</td><td>• Part A: Run time -> Golden pose
• Part B: Run time -> Golden Pose</td><td>Arbitrary pose depending on user</td></tr>
</table>

Golden Pose

Golden pose is the target pose that run time part should be aligned to in alignment or assembly applications, it is consisted of a set of feature locations. Golden pose is saved in train time and would be used in run time pose computation. There are three ways to train a golden pose:

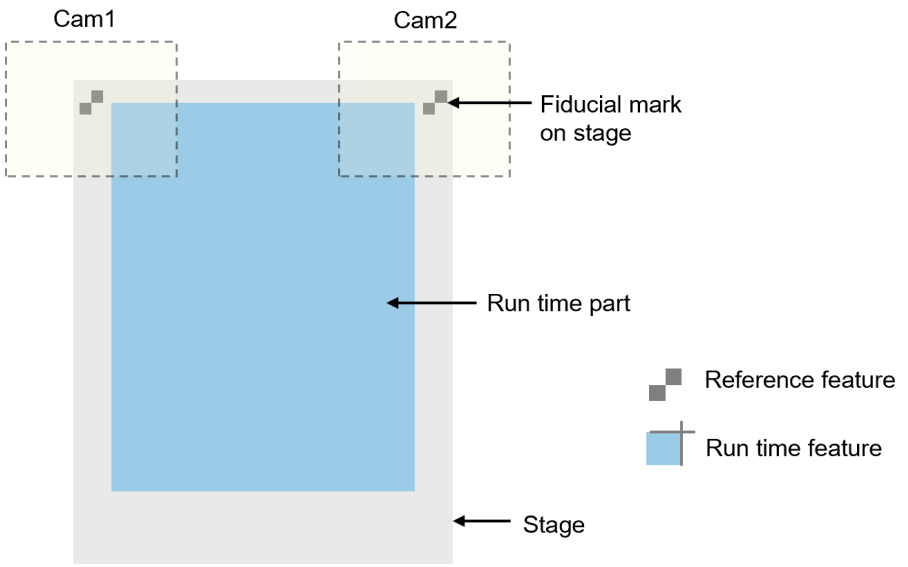
Use current part's feature locations

This method use a run time part to a reference part, extract features of it and save their locations as golden pose. The advantage of it is the train time and run time features are the exactly the same feature types that the vision system doesn't need to set different vision tools for train time and run time feature finding.

Use reference feature locations

Reference feature locations can be used when golden pose needs to be fixed based on certain features on the mechanical device where the run time part rests.

In the example shown below, golden pose is trained using fiducial marks on stage. While run time features are two corners of a part. This method requires the vision system to configure two sets of vision tools, one for reference feature finding, one for run time feature finding. The benefit of this method is that the golden pose would not change even run time part type is changed, thus it maintains the consistency of mechanical operation when it comes to align or assembly.



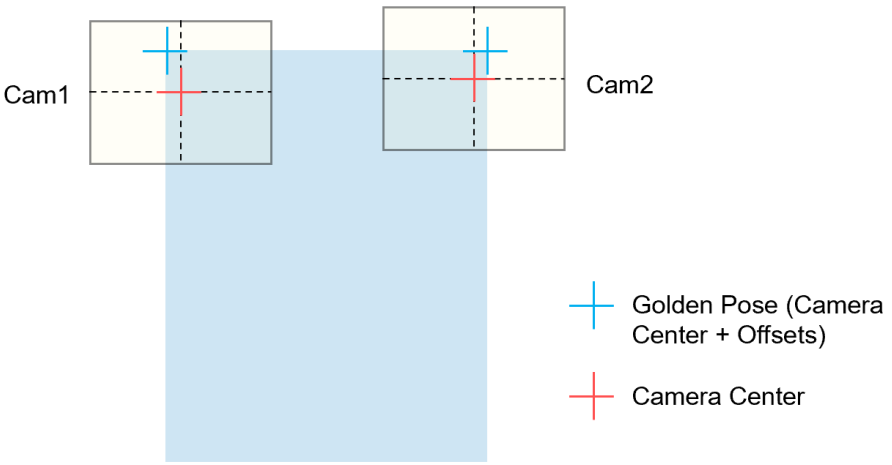
Use camera center

Using camera center as golden pose is another way to keep golden pose fixed when part model changes without camera repositioning. Besides this, using camera center doesn't require extra vision tools for train time feature finding, which is as convenient as using current part's feature locations. However, since camera centers may not always be the ideal physical golden pose where run time part should align to, certain offsets needs to be added to camera centers so that it meets align requirement.

Here is an example:

The Camera1 and Camera2 are installed mechanically not along the same direction with x or y direction of the stage, and golden poses are at the upper part of field of view of cameras so that longer edge features can be captured. Thus the centers of two cameras can’t be directly used as golden poses, offsets should be added to camera centers to set as golden poses.

	Left Point	Right Point
x	Cam1_CenterX + OffsetX1	Cam2_CenterX + OffsetX2
y	Cam1_CenterY + OffsetY1	Cam2_CenterY + OffsetY2



Camera center and offset values are all based on the same shared coordinates with run time features'(Home2D). AlignPlus provides function that allows user add offsets based on the gantry coordinates where cameras are amount to and automatically convert them to coordinates in Home2D.

Part Pose Change Computation

Part position change computation is the first step of Pose Computation. The purpose of it is to calculate the change in pose of the run-time part from the target position.

For different applications, the target position is defined differently.

In alignment applications, all target positions are the part's trained golden pose.

Application	Motion Move Type	Feature Compare Mode	Part Pose Difference (Calculated based on features)
Alignment	Align To Base	Golden Pose	• Run time -> Golden pose
	Align To Gripper	Golden Pose	

In assembly applications, if the chooses to train the golden pose of each part, the target pose for pose change computation is the golden pose established during train-time.

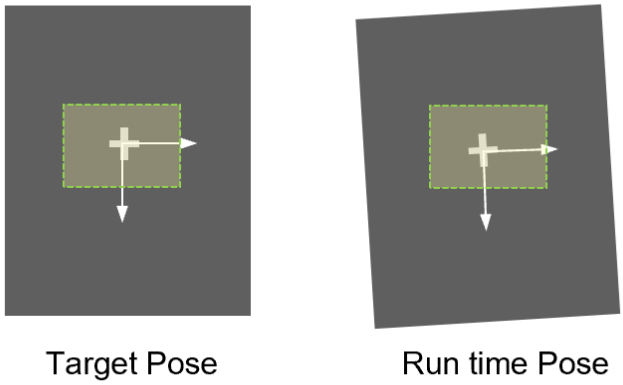
If user choose using paired features, then the target pose is the stationary part's pose (named as Part B)

Application	Motion Move Type	Feature Compare Mode	Part Pose Difference (Calculated based on features)
Assembly	Assembly Blind Transfer	Paired Features	• Part A Run time -> Part B Run time
		Golden Pose	• Part A: Run time -> Golden pose • Part B: Run time -> Golden Pose
	Assembly Guided Pick	Paired Features	• Part A Run time -> Part B Run time
		Golden Pose	• Part A: Run time -> Golden pose • Part B: Run time -> Golden Pose
	Assembly Guided Place	Paired Features	• Part A Run time -> Part B Run time
		Golden Pose	• Part A: Run time -> Golden pose • Part B: Run time -> Golden Pose

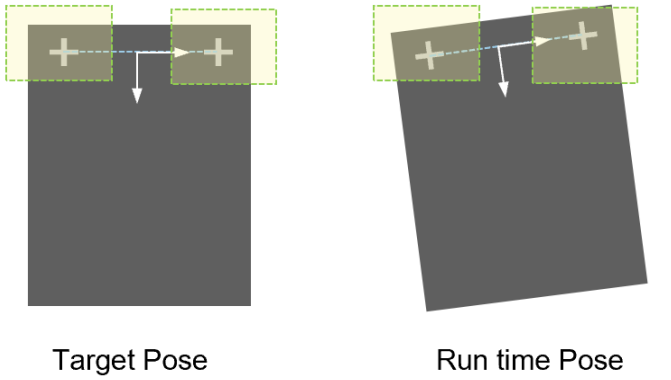
However, irrespective of the definition of the target position, the same techniques are used to compute the change in part pose.

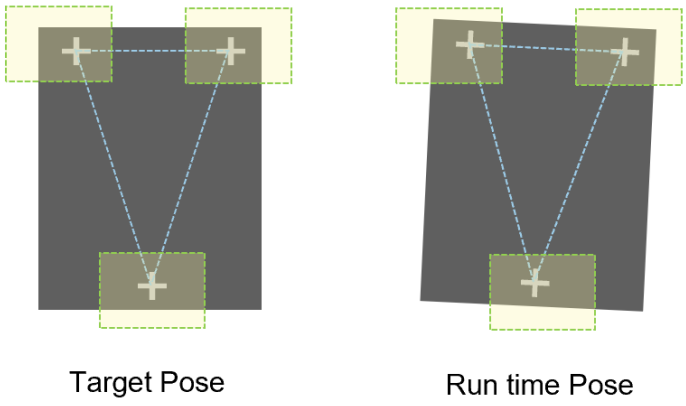
Point Feature Pose Computer

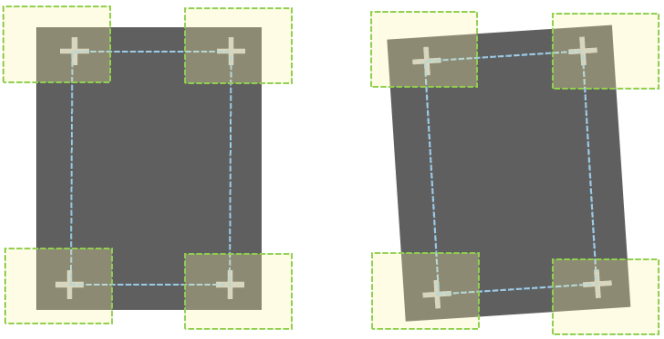
When one has one point feature, the change in part pose is the transform in Home2D that would align the run-time pose (translation and rotation) of the feature to the target pose of the feature.



When more than one point feature is available, the rotation parameter in the pose of each feature is ignored. Only the translation parameters are considered. The pose computer computes an affine transform that maps the run-time points to the train-time points in a fashion that minimizes the distances between the corresponding points in the least squared sense.

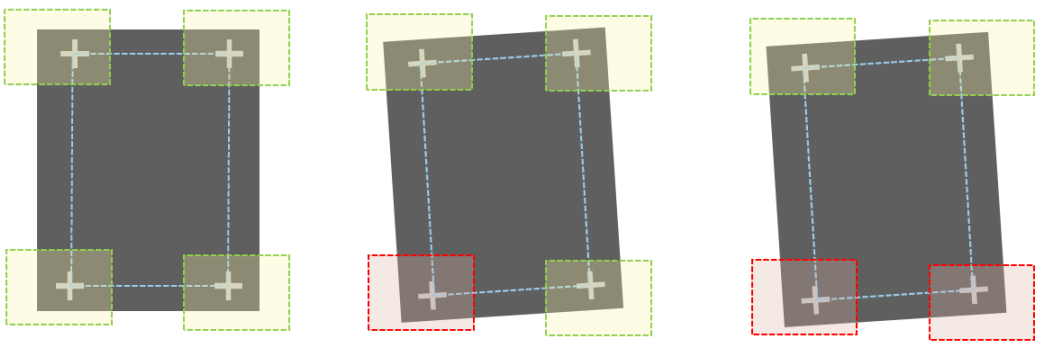






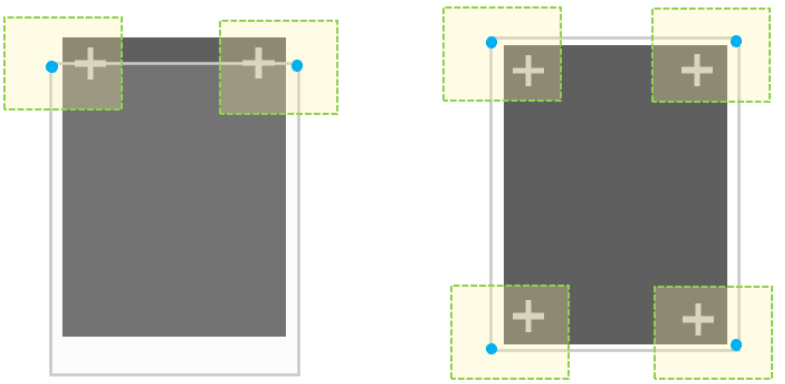
Target Pose Run time Pose

Note:
If one or more features could not be found in the run-time image, the pose computation process will infer the positions of the missing points using the found run-time and train-time features. Once this is done all the points are used to compute the part pose change.



Target Pose Run time Pose => 3-Point Align Run time Pose => 2-Point Align

The result of two-point alignment or assembly could be different with three or four points', here is an example:



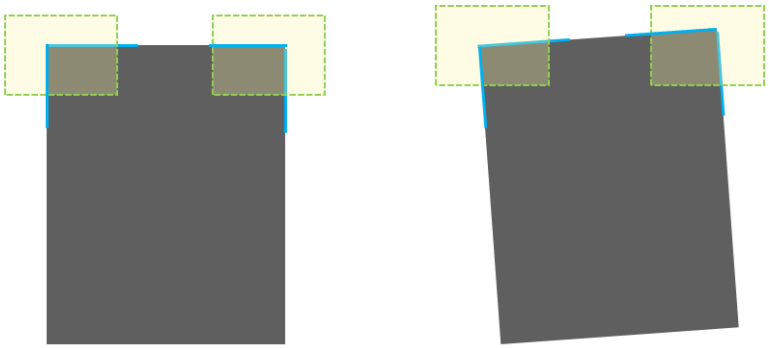
Result of 2-point features Result of 4-point features

When two point features are used, a transform that aligns the two run-time features to the line joining the corresponding train-time features will be computed. When three or more point features are used, the approximate medians of the parts will be aligned.

Line Feature Pose Computer

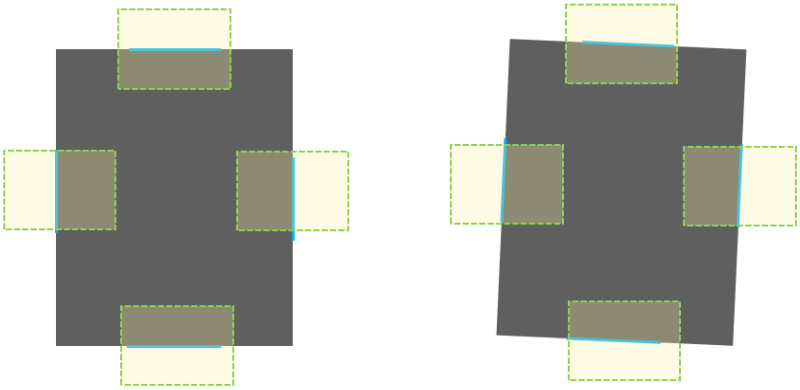
When line feature is used, at least four line features should be found in run time for part pose computation. The four lines could be four edges around two corners or edges along four sides.

- Corner line features



Target Pose Run time Pose

- Side line features

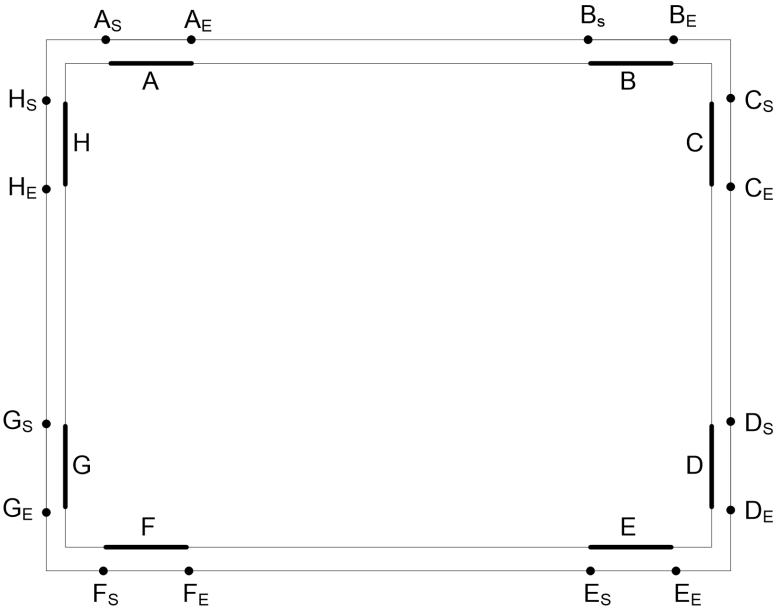


Target Pose Run time Pose

There are two options to center line segments to line segments:

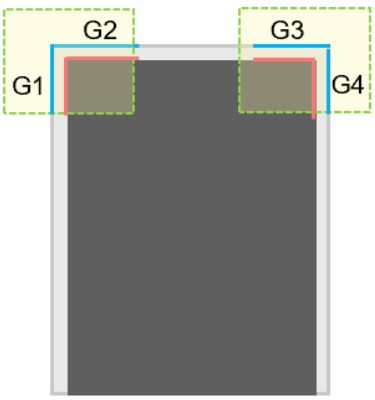
- **CenterGroupALineSegEndPointsToGroupBLines**

  In this mode, the pose computer centers a set of lines on one part to a set of points on another part. Feature correspondences are established between the lines by giving corresponding lines the same names during feature finder setup. This mode will try and minimize the variation in the point-line signed distance across all point-line correspondences.



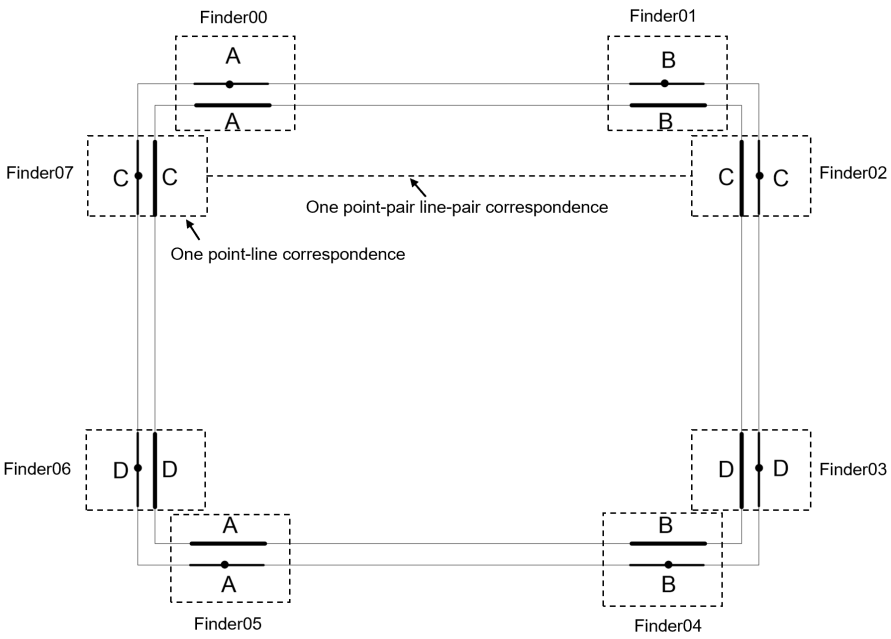
In the diagram above, A, B, C, ..., G are FinderNames of line features. The block uses the end points for the line segments from the first group and the corresponding line segments in the second group to compute the centering transform. In the example, eight line segments are located in each group and sixteen point line pairs are generated, ((A, AS), (A, AE), …, (H, HS), (H, HE)). This mode will try and minimize the variation in the point-line signed distance across all sixteen point-line correspondences.

Here is a result of **CenterGroupALineSegEndPointsToGroupBLines** mode being used in four corner-line assembly: equal values are obtained among all gaps(G1=G2=G3=G4).



- **CenterGroupAPointPairsToGroupBLinePairs**

  In this mode, the feature correspondences are between point pairs and line pairs. The tool will try and minimize the variance in the point-line signed distance difference for each point-pair line-pair correspondence, across all point-pair line-pair correspondences. The point to line distance difference for a given point-pair line-pair correspondence is the difference between the point-line signed distance for the first point and the point-line signed distance for the second point in the point pair.



In the diagram above, A, B, C, D are ProbeID values in line features' UserData dictionary, Finder00 ~ Finder07 are finder names of line features. In this mode, pose computer runs in the following steps:

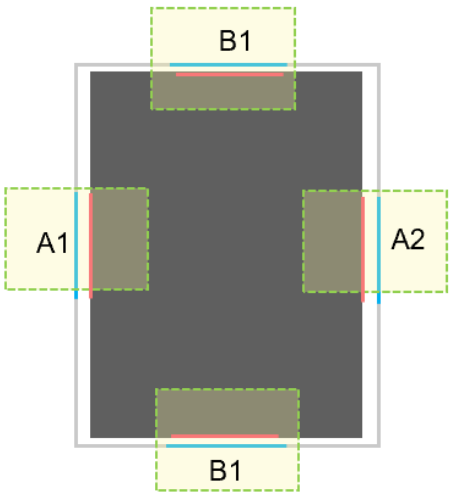
1. Every line segment has a UserData property which is of type CogDictionary. When the computer is configured to work in this mode it expects the dictionary to contain a valid value for the ProbeID key. The block pairs line segments that have the same value for the ProbeID key.

2. It finds the midpoint of all the paired line segments and computes a list of point pairs.

3. For each point pair from the first group, the block finds the corresponding line pair in the second group. It does this by finding lines with the same FinderName property.

4. Once the list of point pairs and the line pairs are computed, the pose computer computes a centering transform

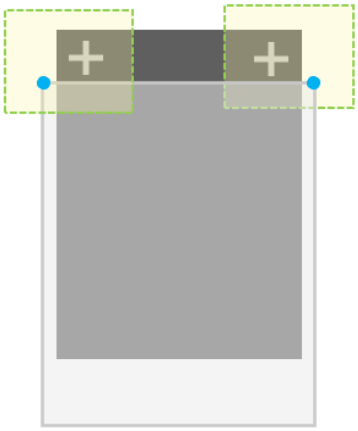
Here is a result of **CenterGroupAPointPairsToGroupBLinePairs** mode being used in four side-line assembly: the symmetric features are paired and equal gaps are obtained(A1=A2, B1=B2).



Custom Pose Computer

Certain application requirements cannot be satisfied by the standard part pose change computers that are described above due to the requirements that they satisfy. In such situations a custom pose computer can be employed.

In the application below, the target features are the two fiducials on a panel, and the run time features are two corners on a polarizer. The requirement is that the two fiducials on the panel should be 2mm above the line joining the two corners on the polarizer.



Custom pose computers allow the application to implement custom assembly rules. It can be implemented in the VisionPro Tool Block that is made available by the application, to use run-time features and target features, and output a transform that satisfies assembly requirement.