COGNEX

AlignPlus® 4.3

Communication Protocol

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.

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

Other product and company trademarks identified herein are the trademarks of their respective owners.

Table of Contents

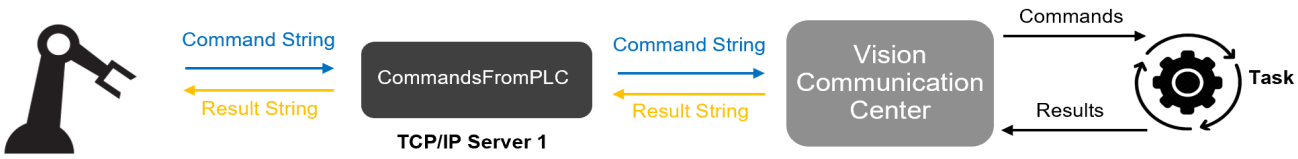
AlignPlus Communication Workflow

In order to execute the various vision tasks in the vision application, command strings, whose format is defined in the Communication Protocol document, can be sent over a TCP/IP channel to the vision application by the customer PLC. Alternatively, other forms of communication can be used. However, the communication data should be converted into the equivalent AlignPlus command string. In return, the execution results will be sent back to the PLC in the form of strings through the same or a different TCP/IP channel depending on whether the tasks are executed in synchronous or asynchronous mode.

The TCP/IP communication procedures in synchronous and asynchronous mode are described below.

Using Synchronous Mode

In synchronous mode, tasks are executed one by one sequentially, therefore, it takes longer time to finish executing all the tasks compared with asynchronous mode. Synchronous mode is recommended when execution sequence is more important than execution time. For example, hand-eye calibration should be finished before its unified cross calibration starts. Feature training is often executed synchronously since it requires many manual checks. In this mode, all command strings and result strings are sent in and out through the same TCP Server named CommandsFromPLC. The steps are:

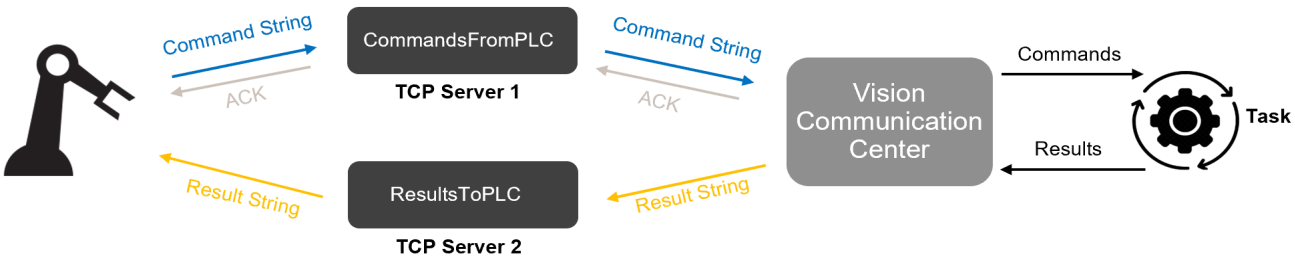


1. A external device sends a command string to CommandsFromPLC.
2. The vision system decodes the command string to identify which task to call, and which execution mode to use.
3. The vision system sends an acknowledge string back to CommandsFromPLC so that the external device knows the command has been received.
4. The vision system runs the requested task and gets the results.
5. The vision system compiles the results into a string and sends it back to CommandsFromPLC
6. The external device receives the result string from CommandsFromPLC.

Using Asynchronous Mode

In asynchronous mode, independent tasks can run asynchronously to reduce total time consumed by the vision system. Asynchronous mode is recommended during run time when reducing execution time is very important to the application. In this mode, two TCP/IP channels are used for data exchange: CommandsFromPLC receives commands from an external device and sends back the acknowledge string, ResultsToPLC only sends the result string back to the external device. This makes the external device be able to have separated channels to send or receive strings without waiting for each other, thus renders communication smoother.

The steps are:



1. An external device sends a command string to CommandsFromPLC

2. The vision system decodes the command string to identify which task to call, and which execution mode to use.

3. The vision system sends an acknowledge string back to CommandsFromPLC so that the external device knows that the command has been received.

4. The vision system runs the requested task and gets the results.

5. The vision system compiles the results into a string and sends it to ResultsToPLC.

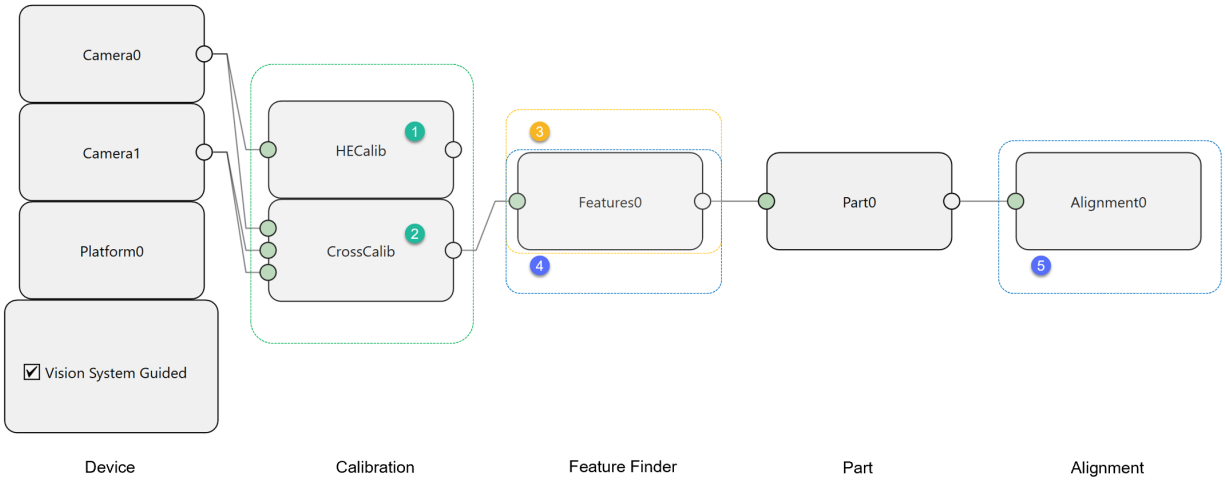
6. The external device receives the result string from ResultsToPLC.

Whether to execute a task in asynchronous mode is encoded in the command string the external device sends to the vision system, together with other information such as task's ID, execution mode, stage's pose, etc. From here on, we will walk through how to get these pieces of information, how to make a command string, and which command should be used for which execution purpose.

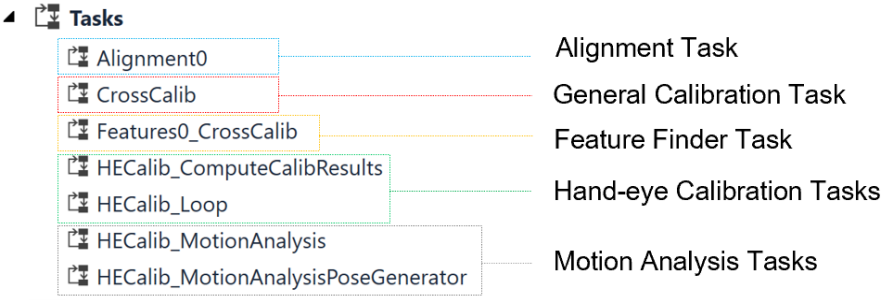
AlignPlus Tasks

AlignPlus tasks are automatically generated by setup wizard based on configurations of calibration, feature finder, and alignment components. Here is an example of setup wizard and its generated tasks.

- Setup Wizard Configuration



- Tasks generated by configuration above



Each component in setup wizard will result in the generation of one or more tasks after setup wizard finishes running. The names of those tasks have certain relationships with their corresponding component names.

Component	Generated Task	Task Example	Task Function	Task Trigger Mode
hand-eye calibration	<Calibration Name>_Loop	HECalib_Loop	The purpose of this task is to generate various motion device poses for the purpose of hand-eye calibration, and at each pose to call the <Calibration Name>_ ComputeCalibResults task. This task is only generated when **Vision System Guided** option is checked on in the stage device in setup wizard	Manually or by external command
	<Calibration Name>_ ComptueCalibResults	HECalib_ ComptueCalibResults	Move the motion device if necessary, capture images, extract features, accumulate features the of calibration target at each stage pose and finally run calibration calculation.	Manually, by external command, or by hand-eye calibration loop task.
General Calibration(Manual Calibration, Checkerboard Calibration , or Cross Calibration)	<Calibration Name>	CrossCalib	Capture images at each acquisition position, and then extract features and run calibration.	Manually, or by external command
Finder	<Finder Name>_<Connected Calibration Name>	Features0_CrossCalib	This task captures all the images needed to find the features on a part and locates all the features by running feature finders. This task can run during train time or run time.	Manually, or by external command
Alignment	<Alignment Name>	Alignment0	Compute the alignment pose for the motion device to achieve the desired assembly characteristics. This task is executed after the tasks that locate the features on the part are executed.	Manually, or by external command

Component	Generated Task	Task Example	Task Function	Task Trigger Mode
Hand-eye Calibration	<Calibration Name>_ MotionAnalysisPoseGenerator	HECalib_ MotionAnalysisPoseGenerator	Motion analysis tasks are used to conduct various tests of the motion device. The purpose of this task is to generate the various motion device poses for the purpose of testing, and at each pose to call the <Calibration Name>_ MotionAnalysis task.	Manually
	<Calibration Name>_ MotionAnalysis	HECalib_MotionAnalysis	This task moves the motion devices, acquires images, extracts features, accumulates features and stage poses, and computes the results using the accumulated data.	Called by Motion Analysis Pose Generator task

Note: When an application that controls the motion device during calibration is generated by selecting the Vision System Guided option, during hand-eye calibration, the PLC has to send commands that execute the tasks that perform "looping". This task calls the <Calibration Name>_ComputeCalibResults task. If the Vision System Guided option is not selected, the PLC has to send commands that execute the <Calibration Name>_ComputeCalibResults task. This condition also applies to tasks that perform motion analysis.

Task Execution Mode

The various vision tasks in the application serve various purposes. Vision tasks can be generated to perform hand-eye calibration, cross-calibration, extract features on parts, to compute alignment parameters, etc. Tasks can be executed in various modes. For example, in an application that employs a single camera to capture two images of non-overlapping image regions on a single part, the task can be executed to acquire images of a part at a first position, and then it can be executed to acquire images of the part at a second position and to use both of the images to locate features. Largely each vision task can be executed in three modes.

- Acquire Only
- Acquire and Process
- Process Image.

The execution mode is defined in the command string that the external device sends to the vision system.

For a typical hand-eye calibration task, If it is in the beginning or end of calibration, the task runs in **Process Image** mode only. If it's in the middle of the loop, the task should acquire images, extract features, and accumulate features at each stage pose so it runs in **Acquire and Process** mode.

Position	Execution Mode
Hand-eye Calibration start or end	Process Only
At each stage pose (in the middle of the calibration loop)	Acquire and Process

General calibration and feature finder tasks share similar patterns in execution mode: when the task has shuttling camera or shuttling part, the task should be called with different actions at each acquisition position: acquire images at non-last positions(**Acquire Only**) , acquire images and run calibration or feature finding at last position(**Acquire and Process**). If the task has no shuttling camera or shuttling part, then it should be called only once in **Acquire and Process** mode.

Position	Execution Mode
First position, second position, ...	Image Acquire
Last position	Image Acquire and Process

An alignment task does not acquire any images, so it should always be called with **Process Image** mode.

StepID

StepIDs are automatically defined when the application is generated by the setup wizard. They can be found at the top of CommandHandler script.

```
/* Communication Info Begin
StepID = 0                    HECalib_Loop
StepID = 1                    HECalib_ComputeCalibResults, Camera position: 0
StepID = 2                    CrossCalib, Camera position: 0
StepID = 3                    CrossCalib, Camera position: 1
StepID = 4                    CrossCalib, Camera position: 2
StepID = 5                    Features0, Calibration: CrossCalib, camera position: 1
StepID = 6                    Features0, Calibration: CrossCalib, camera position: 2
StepID = 7                    Alignment0
StepID = 8                    HECalib_MotionAnalysis
StepID = 9                    HECalib_MotionAnalysisPoseGenerator, Camera position: 0
   Communication Info End */
```

For the sample application above, the automatically generated StepIDs are as follow:

- Hand-eye calibration loop task's step ID is 0
- Hand-eye calibration ComputeCalibResult task's step ID is 1
- Cross calibration task has three StepIDs (2, 3, 4) since it has three different image acquisition positions. Accordingly, external devices should call this task three times with different StepIDs and different execution modes to finish the cross calibration process
- Features0 has two StepIDs (5, 6) since it acquires images at two different positions. It should be called two times with different StepIDs during both train-time and run-time
- Alignment0 has one StepID, 7
- Motion analysis tasks have stepIDs 8 and 9.

EncodedID

EncodedID is part of the command string that an external device sends to the vision system to run a task. It contains the StepID of requested task, as well as the task execution mode:

1. Acquire Only

In this mode, EncodedID = StepID + 1000.

2. Acquire and Process

In this mode, EncodedID = StepID

3. Process Only

In this mode, EncodedID = StepID + 2000.

This table summarizes the various vision tasks that can be performed by the sample application:

Category	Task	StepID	Action	EncodedID
Calibration	HECalib_ Loop	0	Only one command with **Process Image** action to trigger hand-eye calibration loop, then the loop will take control the rest of process	2000
	CrossCalib	2	First command to **Acquire Image** at position 0	1002
		3	Second command to **Acquire Image** at position 1	1003
		4	Third command to **Acquire Image and Process** at position 2	4
Train Time	Features0_ CrossCalib	5	First command to **Acquire Image** at position 0	1005
		6	Second command to **Acquire Image and Process** at position 1	6
Run Time	Features0_ CrossCalib	5	First command to **Acquire Image** at position 0	1005
		6	Second command to **Acquire Image and Process** at position 1	6
	Alignment0	7	One command to **Process Image**	2007

Command Structure

Command String Structure

Command String is a string which external devices send to the vision application requesting one task or multiple tasks be executed. Command String is composed of three parts: Command Key, EncodedID and optional parameters. The three parts are joined using comma without space. Each individual optional parameter is also linked with each other using comma without space.

<CommandKey>,<EncodeID>[,<TargetID>][,<PartID>][,<ResultMode>][,<X>,<Y>,<Z>,<A>,<B>,<C>][,<UserString>]

Optional depending on CommandKey

Command Key

Command key indicates the type of action that a vision task has to perform. For example, the hand-eye calibration task can be started, continued or completed by sending a suitable command key. Similarly, a task that finds the features on a part can be executed to perform train-time or run-time actions through a suitable command key.

Here is all the available command keys in AlignPlus:

CommandKey	Description	When to use
HEB	Hand-eye Calibration Begins	Motion Guided Hand-eye Calibration
HE	Hand-eye Calibration	
HEE	Hand-eye Calibration Ends	
ACB	Auto Calibration Begins	Vision Guided Hand-eye Calibration
AC	Auto Calibration	
IC	Intrinsic Calibration	Checkerboard Calibration, Cross Calibration or Manual Calibration
TA	Train Alignment (registers the golden pose of the target)	Train Time Commands
TT	Train VGR step (register target, single or multiply shots)	
TTR	Train VGR step (register robot pick/place position for a target)	
LF	Locate Features	Feature Finding in run time.
GP	Get Pose	Pose Computing
LFA	Locate Features Asynchronously	Feature Finding in run time in asynchronous mode
GPA	Get Pose Asynchronously	Pose Computing in asynchronous mode
LFGP	Locate Features, then run Get Pose	Feature Finding and Pose Computer in run time
MEA	Measure	Run inspection
MEAA	Measure Asynchronously	Run inspection asynchronously
LFMEA	Locate Features, then Measure	Run feature finding first, then run inspection
MGP	Multi-part Get Pose	Multi-part pose computation
MGPA	Multi-part Get Pose Asynchronously	Run multi-part pose computation
LFMGP	Locate Features, then run Multi-part Get Pose	Run feature finding first, then run multi-part pose computation synchronously

EncodedID

See definition of encodedID in AlignPlus Tasks on page 6.

Parameters

Some commands require additional parameters from the PLC. For example, applications such as LF, or GP require the current pose of the alignment device to execute correctly. These additional parameters are encoded in the command string.

Here is the summary of all possible parameters in AlignPlus commands; different commands will have different combination of these parameters.

Parameters	Description	When to use
TargetID	0 (not used)	Hand-eye Calibration
PartID	Serial number of part if it has, or any string to identify a part.	Feature Finding Run Time and Pose Computer
ResultMode	Abs: using absolute position as return result from vision to motion device Off: using relative position as return result from vision to motion device	Pose Computer
X	The X-coordinate of the current position of the motion system (x in 2D plane)	Hand-eye Calibration, Feature Finding train time and run time, Pose Computing
Y	The Y-coordinate of the current position of the motion system (y in 2D plane)	
Z	The Z-coordinate of the current position of the motion system	
A	The Z-rotation of the current position of the motion system (theta in 2D plane)	
B	The Y-rotation of the current position of the motion system	
C	The X-rotation of the current position of the motion system	
UserString	The optional field which can be used to send some customized information to AlignPlus program (certain scripting needs to be done to utilize this self-defined string).	Feature Finding run time
RequestData	Whether return measurement result values besides their OK/NG information	Inspection
StatusOnly	Whether return target X,Y,Theta for each sub-region besides their OK/NG information	Multi-part alignment
Index	Index of sub-region in which run time part should be aligned	

Acknowledge Signal

Once the vision system receives a run time command (such as LF, GP commands) from an external device, it will generate a **Token** for that command and send it back to the external device in the form of a string. The string format is merely one parameter: Token, followed by CR/LF.

<Token>

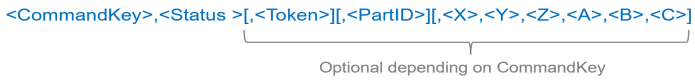
Token

A token is a unique number automatically generated by an AlignPlus program as a response for a received command. The same token will be returned in the result string again when the requested tasks finish execution. Therefore, the external device can use the token to match the result string with the command it sent out. This is especially important when the sequence of received result strings are not the same with the sequence of their corresponding sent-out commands due to running in the asynchronous mode.

Result String Structure

Result string is a string that the vision system sends to an external device after the task requested by the command from that external device finishes execution. One command will have one and only one result string in return. Result string is composed of three parts: Command Key, Status and Parameters, and in addition is followed by CR/LF at the end.

These three parts and all individual parameters within **Parameters** are joined with comma without space.



Command Key

CommandKey of result string is always the same with the Commandkey in its corresponding command string.

Status

In most cases, Status only indicates whether the requested task runs successfully. In a few other cases, Status is also used to indicate whether a process is finished or not (such as used in ACB/AC commands).

Value	Description
ErrorCode on page 14	The requested vision task fails
1	The requested vision task runs successfully and that is the end of the current procedure
2	The requested vision task runs successfully and the current procedure is to be continued

Parameters

Some commands require the vision system to return additional results besides status. For example, GP commands require the target pose of stage be returned by the vision system to guide the stage's move. These additional results are encoded in the result string in the form of individual parameters.

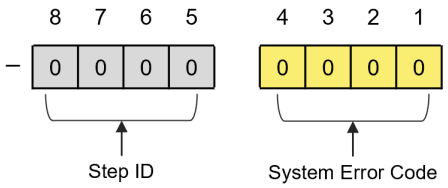
Here is the summary of all possible parameters in result strings; different commands request different combinations of these parameters.

Parameters	Description	When to use
Token	The Token identifying the corresponding command.	Feature Finding Run Time and Pose Computer
PartID	The string showing ID of each multi-step sequence, i.e., all commands in the same alignment task must have the same PartID.	
X	The X-coordinate of the destination position of the motion system (x in 2D plane)	Hand-eye Calibration, Feature Finding train time and run time, Pose Computing
Y	The Y-coordinate of the destination position of the motion system (y in 2D plane)	
Z	The Z-coordinate of the destination position of the motion system	
A	The Z-rotation of the destination position of the motion system (theta in 2D plane)	
B	The Y-rotation of the destination position of the motion system	
C	The X-rotation of the destination position of the motion system	

Parameters	Description	When to use
ResultStatus	Whether Measurement results are within specifications	Measurement and gauging
DataCount	The count of measurement results	
Data[N]	The N-th measurement result data.	
ResultCount	The count of alignment results	Multi-Part Pickup
Status[N]	Whether the N-th sub-region's pose computer runs successfully	

ErrorCode

When a requested task fails, an error code indicating which task fails and why it fails will be encoded into a negative integer in the result string and returned to the external device. Error Code consists of two parts: StepID (the upper four digits) and System Error Code (the lower four digits).



StepID is the stepID of requested task. System Error Code is a numbered code that determines the nature of an error and why it occurs.

Here is the table of System Error Code:

Error Type	System Error Code
None	-9999
TimeOut	-1000
UnKnownCommand	-1001
TooFewArguments	-1003
InvalidArgumentType	-1004
InvalidArgument	-1005
CommandNotAllowed	-1006
Busy	-1008
NotSupported	-1010
AcquisitionFailed	-1014
NotCalibrated	-2001
CalibrationFailed	-2002
InvalidCalibrationData	-2003
FeatureNotTrained	-3001
FeatureNotFound	-3002
LCheckFailed	-3006

ErrorCode is computed by the following equation:

ErrorCode = StepID * -10000 + System Error Code

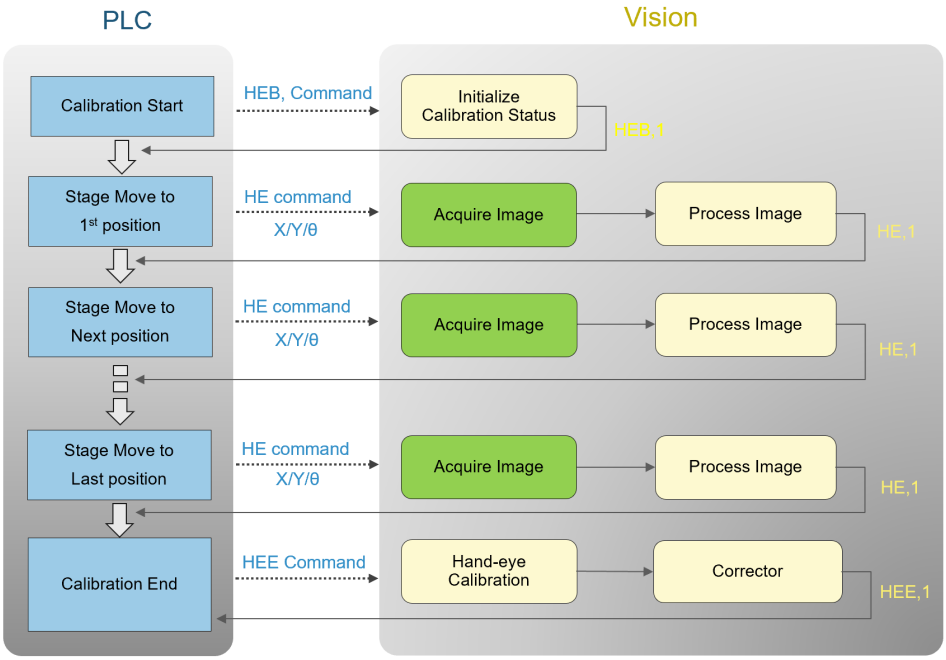
For example, ErrorCode '-42003' means the error occurs in the task whose stepID is 4, and the cause of the error is that the task's calibration data is invalid.

Commands

Calibration

Motion Guided Hand-eye Calibration Commands

In motion guided hand-eye calibration, the motion system is responsible for initializing a hand-eye calibration, computing the next target pose and triggering the vision system after stage finishes moving to the target pose, and ending the hand-eye calibration when all poses are run through. Whereas the vision system acts in response to the motion system's commands.



HEB

HEB stands for hand-eye calibration begins, it initiates a motion guided hand-eye calibration.

Command String

HEB,<EncodedID>

Field	Arguments
<EncodedID>	EncodedID = StepID of hand-eye calibration task + Process Tag(2000)

Result String

HEB,<Status>CR/LF

Field	Arguments
<Status>	The result status 1 : Success ErrorCode: Fail

HE

Executes a hand-eye calibration task to acquire images and accumulate extracted features of calibration target at a certain stage pose.

Command String

HE,<EncodedID>,<TargetID>,<X>,<Y>,<Z>,<A>,<B>,<C>

Field	Description
<EncodedID>	EncodedID = StepID of hand-eye calibration task
<TargetID>	0 (not used)
<X>	The X-coordinate of the current position of the motion system
<Y>	The Y-coordinate of the current position of the motion system
<Z>	0 (not used)
<A>	The theta(in Degree) of the current position of the motion system
<B>	0 (not used)
<C>	0 (not used)

Result String

HE,<Status>CR/LF

Field	Arguments
HE	Command Key
<Status>	The result status 1 : Success ErrorCode : Fail

HEE

HEE stands for hand-eye calibration end. It triggers the vision system to compute the hand-eye calibration results based on the data collected in the previous “HE” steps.

Command String

HEE,<EncodedID>

Field	Arguments
<EncodedID>	EncodedID = StepID of hand-eye calibration task + Process Tag(2000)

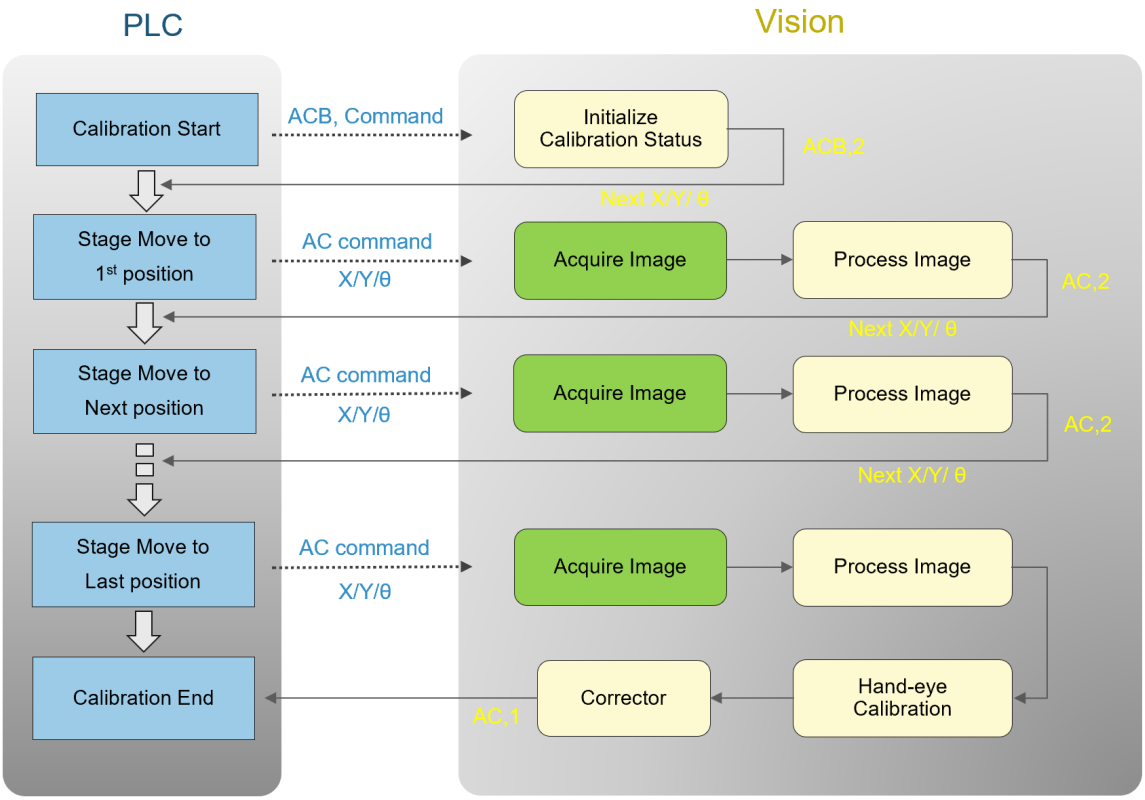
Result String

HEE,<Status>CR/LF

Field	Arguments
<Status>	The result status 1 : Success ErrorCode : Fail

Vision Guided Hand-eye Calibration Commands

In vision guided hand-eye calibration, the motion system acts in response to the vision system's result string after a hand-eye calibration begins: after receiving hand-eye calibration start command, the vision system is responsible for computing the next target pose and send it back to the motion system. In response, motion system would move the stage to the requested target pose and return a command to the vision system with updated stage pose. When all poses are walked through, the vision system will run hand-eye calibration computation and then send an end status in the result string to the motion system, so that the motion system knows the hand-eye calibration is finished.



ACB

ACB stands for auto calibration begins. It initiates a vision guided hand-eye calibration.

Command String

ACB,<EncodedID>

Field	Arguments
<EncodedID>	EncodedID = StepID of hand-eye calibration loop task + Process Tag(2000)

Result String

When requested task runs successfully:

ACB,<Status>, <X>,<Y>,<Z>,<A>,<B>,<C>CR/LF

Field	Arguments
<Status>	2 : Success, return the next position
<X>	The X-coordinate of the next position for the motion system
<Y>	The Y-coordinate of the next position for the motion system
<Z>	0 (not used)
<A>	The theta-value(in Degree) of the next position for the motion system
<B>	0 (not used)
<C>	0 (not used)

When fails:

ACB,<Status>

Field	Arguments
<Status>	ErrorCode : Fail

AC

Executes a hand-eye calibration task to acquire images and accumulate extracted features of the calibration target at a certain stage pose.

Command String

AC,<EncodedID>,<TargetID>,<X>,<Y>,<Z>,<A>,<B>,<C>

Field	Description
<EncodedID>	StepID of hand-eye calibration loop task
<TargetID>	0 (not used)
<X>	The X-coordinate of the current position of the motion system
<Y>	The Y-coordinate of the current position of the motion system
<Z>	0 (not used)
<A>	The theta(in Degree) of the current position of the motion system
<B>	0 (not used)
<C>	0 (not used)

Result String

When the whole hand-eye calibration process is to be continued and the current calibration step runs successfully:

AC,<Status>,<X>,<Y>,<Z>,<A>,<B>,<C>CR/LF

Field	Arguments
AC	Command Key
<Status>	2 : Success, return the next position
<X>	The X-coordinate of the next position of the motion system
<Y>	The Y-coordinate of the next position of the motion system
<Z>	0 (not used)
<A>	The theta(in Degree) of the next position of the motion system
<B>	0 (not used)
<C>	0 (not used)

When the whole hand-eye calibration process is to be continued but current calibration step fails:

Field	Arguments
AC	Command Key
<Status>	ErrorCode

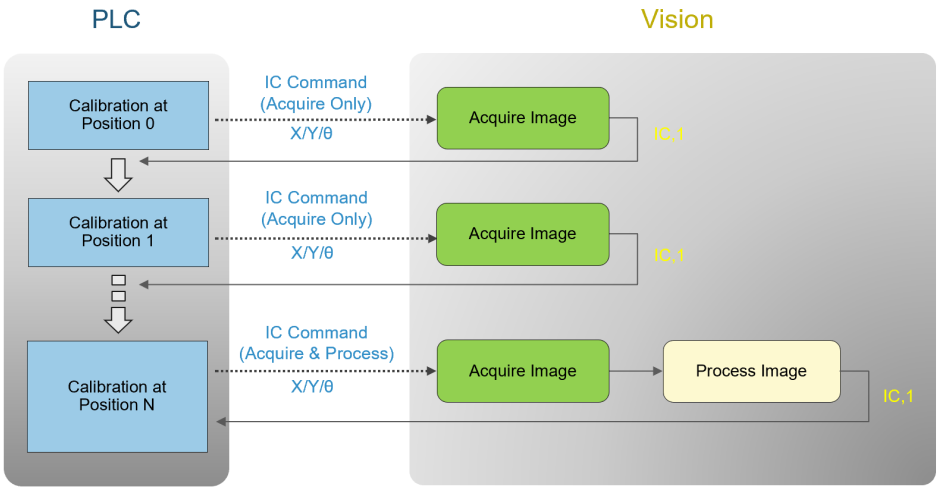
When the whole hand-eye calibration process is finished:

AC,<Status>CR/LF

Field	Arguments
AC	Command Key
<Status>	The result status 1 : Hand-eye calibration is done successfully ErrorCode : Hand-eye calibration fails

General Calibration Commands

General calibrations are calibrations besides hand-eye calibration, such as cross calibration, manual calibration, or checkerboard calibration. General calibrations share the same calibration process and commands as displayed below: The external device will send one or more IC commands to the vision system depending on the number of acquisition positions that the calibration has. Execution mode for non-last acquisition position of the calibration should be set as Acquire Only and encoded in IC command. For the last acquisition position, the execution mode should be changed to Acquire and Process to acquire image, execute calibration computation, and end the calibration process.



IC

IC stands for intrinsic calibration. It is used to trigger general calibration.

Command String

IC,<EncodedID>,<X>,<Y>,<Z>,<A>,<B>,<C>

Field	Arguments
<EncodedID>	• For non-last position: EncodedID = StepID of calibration task at current position + Acquire Tag (1000) • For last position: EncodedID = StepID of calibration task at last position
<X>	The X-coordinate of the current position of the motion system
<Y>	The Y-coordinate of the current position of the motion system
<Z>	0 (not used)
<A>	The theta (in Degree) of the current position of the motion system
<B>	0 (not used)
<C>	0 (not used)

Result String

IC,<Status>CR/LF

Field	Arguments
<Status>	The result status 1 : Success ErrorCode : Fail

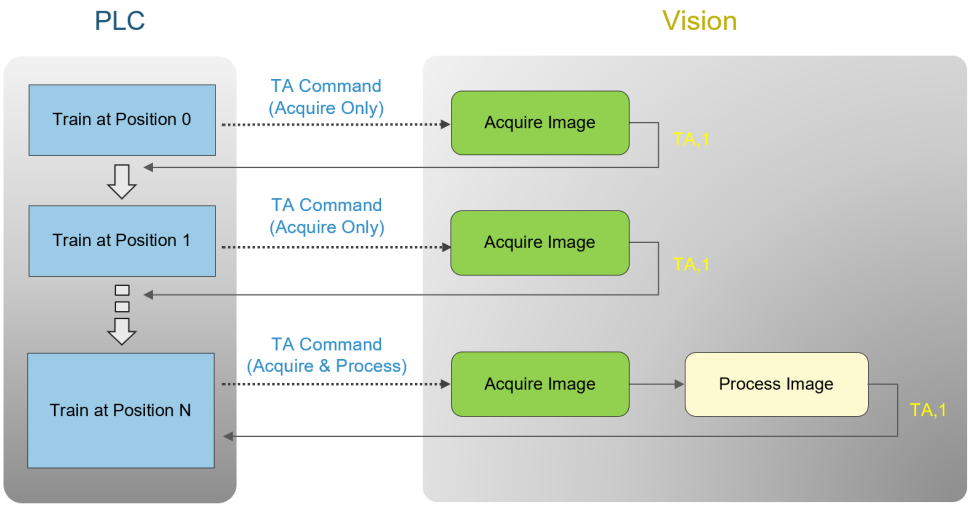
Train Time

Golden Pose Training Commands

Golden pose training is a necessary process before run time feature finding for alignment applications and assembly applications where golden pose mode is used. Depending on whether the gripper's pick or place pose should be registered at the same time or not, TT or TA command can be used to train golden pose.

TA

Trains golden pose for target feature finding task. The external device should send one or more TA commands depending on the number of acquisition position of the feature finding task. At non-last acquisition positions, the execution mode should be set as Acquire Only and encoded in the TA commands; at the last acquisition position, the execution mode should be changed to Acquire and Process instead to acquire the last set of images and execute the feature extraction.



Command String

TA,<EncodedID>

Field	Arguments
<EncodedID>	• For non-last position: EncodedID = StepID of feature finder task at current position + Acquire Tag (1000) • For last position: EncodedID = StepID of feature finder task at last position

Result String

TA,<Status>CR/LF

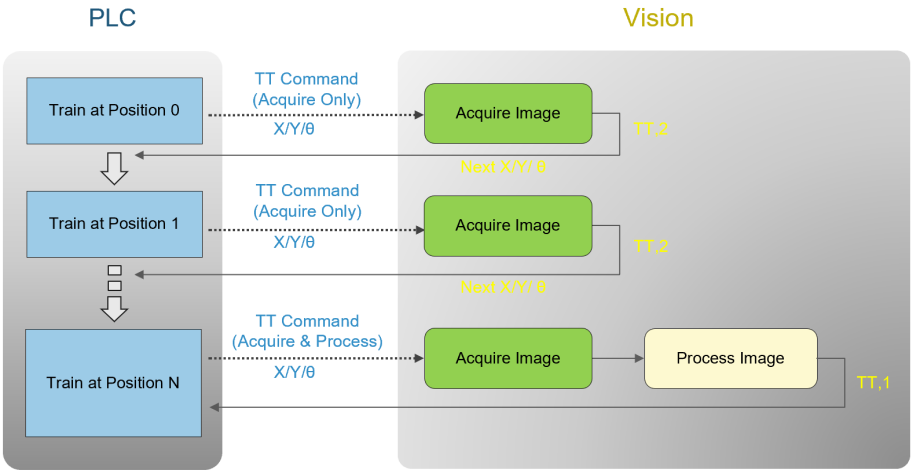
Field	Arguments
<Status>	The result status 1 : Success ErrorCode : Fail

Note: Before vision tools in features finder are configured, TA commands can be used to trigger image acquisition only.

TT

Trains golden pose and registers the target feature for VGR(Vision Guide Robot) application. Similar with TA command, one or more TT commands should be sent to the vision system to finish all the image acquisition and then process feature finding

at the last position. After all TT commands finish running, a TTR command should be followed to register the robot's pick or place pose.



Command String

TT,<EncodedID>,<X>,<Y>,<Z>,<A>,<B>,<C>

Field	Arguments
<EncodedID>	• For non-last position: EncodedID = StepID of feature finder task at current position + Acquire Tag (1000) • For last position: EncodedID = StepID of feature finder task at last position
<X>	The X-coordinate of the current position of the motion system
<Y>	The Y-coordinate of the current position of the motion system
<Z>	0 (not used)
<A>	The theta(in degrees) of the current position of the motion system
<B>	0 (not used)
<C>	0 (not used)

Result String

TT,<Status>CR/LF

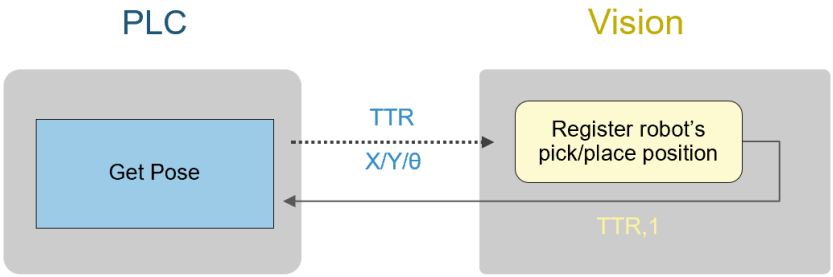
Field	Arguments
<Status>	The result status 1 : Success ErrorCode : Fail

Motion Train Pose Command

In AlignToGripper applications, gripper's train-time pick pose needs to be registered to the vision system, so that during run time, this pose can be used as an input for the vision system to compute target poses for the gripper. Likewise, when golden pose mode is used, AssemblyGuidedPick and AssemblyGuidePlace applications needs to register gripper's train-time pick and place poses for run time pose computation.

TTR

Registers gripper's pick or place pose for a VGR application. It should be called after TT commands finish running.



Command String

TTR,<EncodedID>,<X>,<Y>,<Z>,<A>,<B>,<C>

Field	Arguments
<EncodedID>	EncodedID = StepID of feature finder task at last position + Process Tag(2000)
<X>	The X-coordinate of the robot's pick (or place) pose
<Y>	The Y-coordinate of the robot's pick (or place) pose
<Z>	0 (not used)
<A>	The theta of the robot's pick (or place) pose
<B>	0 (not used)
<C>	0 (not used)

Result String

TTR,<Status>CR/LF

Field	Arguments
<Status>	The result status 1 : Success ErrorCode : Fail

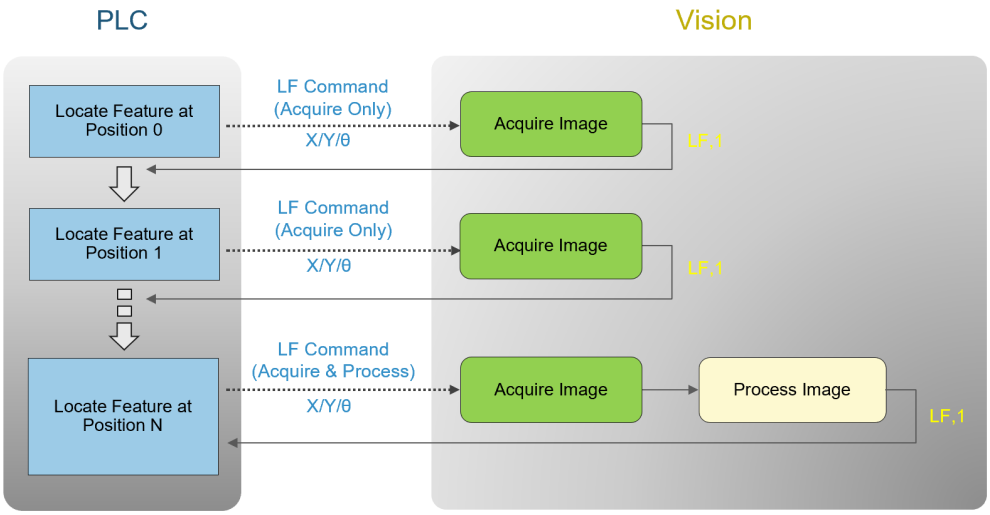
Run Time

Feature Finding Commands

Feature finding commands are used to trigger feature finding task in run time.

LF

Locates features of a part in synchronous mode. The external device should send one or more LF commands depending on the number of acquisition position of the feature finding task. At non-last acquisition positions, the execution mode should be set as Acquire Only and encoded in the LF commands; at the last acquisition position, the execution mode should be changed to Acquire and Process instead to acquire the last set of images and execute the feature extraction.



Command String

LF,<EncodedID>,<PartID>,<X>,<Y>,<Z>,<A>,<B>,<C>[,<UserString>]

Field	Arguments
<EncodedID>	• For non-last position: StepID of feature finder task at current position + Acquire Tag(1000) • For last position: StepID of feature finder task at last position
<PartID>	Serial number of part if it has, or any string if it is consistent for one part (PartID used for multiple LF commands should be the same if they are used for the same part)
<X>	The X-coordinate of the current position of the motion system
<Y>	The Y-coordinate of the current position of the motion system
<Z>	0 (not used)
<A>	The theta(in degrees) of the current position of the motion system
<B>	0 (not used)
<C>	0 (not used)
[,<UserString>]	Optional, for user to customize some extra data

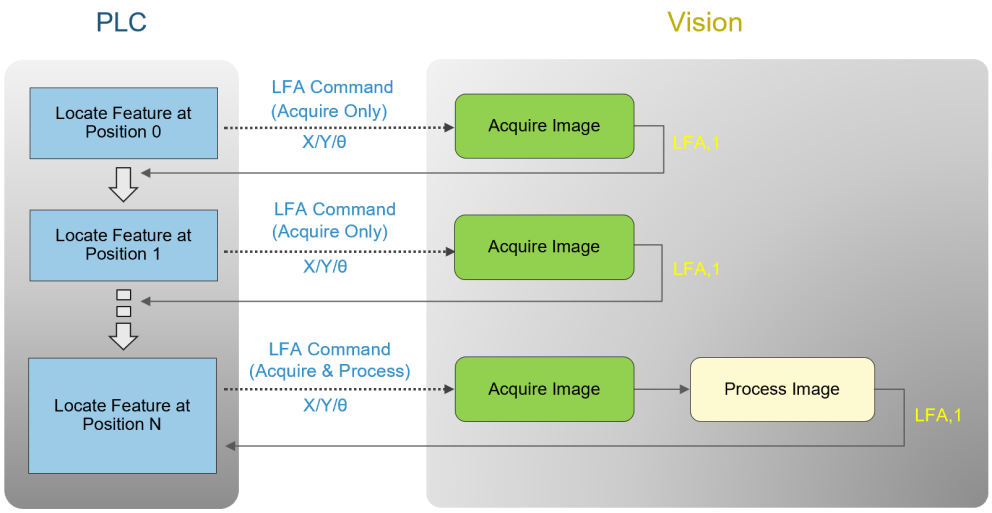
Result String

LF,<Status>,<Token>,<PartID>CR/LF

Field	Arguments
<Status>	The result status 1 : Success ErrorCode : Fail
<Token>	The Token identifying the corresponding command.
<PartID>	The PartID provided by command string

LFA

Locates the features of a part in asynchronous mode. The command sequence and execution modes are the same as described in LF command.



Command String

LFA,<EncodedID>,<PartID>,<X>,<Y>,<Z>,<A>,<B>,<C>[,<UserString>]

Field	Arguments
<EncodedID>	For non-last position: StepID of feature finder task at current position + Acquire Tag(1000) For last position: StepID of feature finder task at last position
<PartID>	Serial number of part if it has, or any string if it is consistent for one part (PartID used for multiple LF commands should be the same if they are used for the same part)
<X>	The X-coordinate of the current position of the motion system
<Y>	The Y-coordinate of the current position of the motion system
<Z>	0 (not used)
<A>	The theta(in degrees) of the current position of the motion system
<B>	0 (not used)
<C>	0 (not used)
[,<UserString>]	Optional, for user to customize some extra data

Result String

LFA,<Status>,<Token>,<PartID>CR/LF

Field	Arguments
<Status>	The result status 1 : Success ErrorCode : Fail

Field	Arguments
<Token>	The Token identifying the corresponding command.
<PartID>	The PartID provided by command string

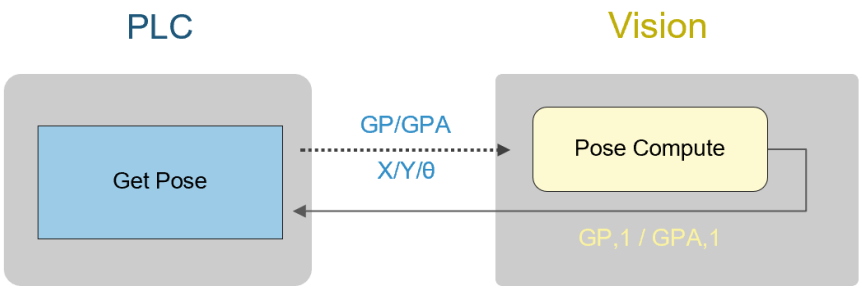
Note:
Before a new LF/LFA is sent to the same feature finding task, the external device needs to make sure the previous LF/LFA command has already finished execution, otherwise AlignPlus program will throw an error saying task is busy. The external device can compare the <Token> in the acknowledge signal and the <Token> in the result string to confirm whether a requested task is finished. If two tokens are the same, it means that the task has already been executed, otherwise, the task is still running.

nPose Computer Commands

Pose computer command is used to trigger the vision system to compute target pose for motion device to move, so that when the part is aligned or assembled, the alignment or assembly characteristics can be achieved. Pose computer command has two types: GP/GPA, MGP/MGPA. GP and GPA are for single part alignment or two parts assembly applications, MGP and MGPA are for multi-part pickup applications. GP and MGP request task to run in synchronous mode, whereas GPA and MGPA request run in asynchronous mode.

GP/GPA

Computes the destination pose from all the feature finding steps(triggered "LF" or "LFA" commands) which share the same PartID. This is the last command in a multi-step alignment sequence, and it assumes all features have been located successfully with preceding “LF”/"LFA" commands.



GP and GPA commands share the same command format. The only difference between them are that GP command will run the requested task synchronously, whereas GPA command runs task asynchronously.

Command string for GP command:

GP,<EncodedID>,<ProductID>,<ResultMode>,<X>,<Y>,<Z>,<A>,<B>,<C>

Command string for GPA command:

GPA,<EncodedID>,<ProductID>,<ResultMode>,<X>,<Y>,<Z>,<A>,<B>,<C>

Field	Arguments
<EncodedID>	Alignment task + Process Tag(2000)
<PartID>	Serial number of part if it has, or any string as long as it is consistent for one part (PartID used here should be the same as used in LF commands if they are from the same part)
<ResultMode>	Abs (absolute value) or Off (offset from the current pose)
<X>	The X-coordinate of the current position of the motion system
<Y>	The Y-coordinate of the current position of the motion system

Field	Arguments
<Z>	0 (not used)
<A>	The theta(in degrees) of the current position of the motion system
<B>	0 (not used)
<C>	0 (not used)
[,<UserString>]	Optional, for user to customize some extra data

Result String

When task runs successfully:

GP/GPA,<Status>,<Token>,<PartID>,<X>,<Y>,<Z>,<A>,<B>,<C>CR/LF

Field	Arguments
<Status>	1
<Token>	The Token identifying the corresponding command
<PartID>	PartID provided by command string
<X>	The X-coordinate of the destination position for the motion system
<Y>	The Y-coordinate of the destination position for the motion system
<Z>	0 (not used)
<A>	The theta(in degrees) of the destination position for the motion system
<B>	0 (not used)
<C>	0 (not used)

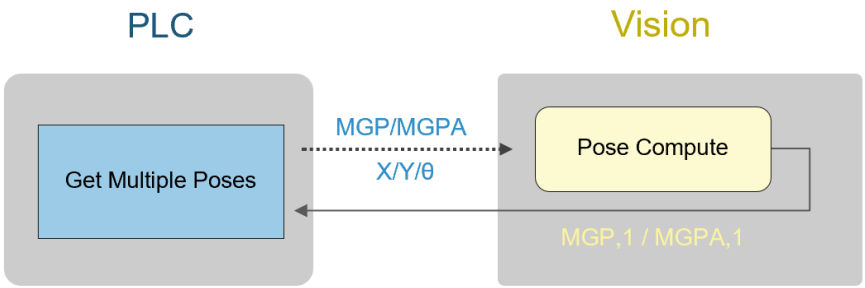
When task fails:

GP/GPA,<Status>,<Token>,<PartID>CR/LF

Field	Arguments
<Status>	ErrorCode
<Token>	The Token identifying the corresponding command
<PartID>	PartID provided by command string

MGP/MGPA

Runs multi-part pickup. This is the last command in a multi-step alignment sequence, and it assumes all features have been located successfully with preceding “LF” commands.



Command string

MGP and MGPA commands share the same command format. The only difference between them are that MGP command will run the requested task synchronously, whereas MGPA command runs task asynchronously.

Command string for MGP command:

MGP,<EncodedID>,<PartID>,<StatusOnly>,<ResultMode>,<X>,<Y>,<Z>,<A>,<B>,<C>,<UserString>,<MainIndex>[,<Index2>]...[,<IndexN>]

Command string for MGPA command:

MGPA,<EncodedID>,<PartID>,<StatusOnly>,<ResultMode>,<X>,<Y>,<Z>,<A>,<B>,<C>,<UserString>,<MainIndex>[,<Index2>]...[,<IndexN>]

Field	Arguments
<EncodedID>	Alignment task + Process Tag(2000)
<PartID>	Serial number of part if it has, or any string if it is consistent for one part (PartID used here should be the same as used in LF commands if they are from the same part)
<StatusOnly>	0: return string feedback OK/NG information as well as target X,Y,Theta for each sub-region. 1: return string only feedback OK/NG for each sub-region
<ResultMode>	Abs (absolute value) or Off (offset from the current pose)
<X>	The X-coordinate of the current position of the motion system
<Y>	The Y-coordinate of the current position of the motion system
<Z>	0 (not used)
<A>	The theta(in degrees) of the current position of the motion system
<B>	0 (not used)
<C>	0 (not used)
,<UserString>	For user to customize some extra data(required field, not optional)
<MainIndex>[,<Index2>]...[,<IndexN>]	Indexes of sub-regions of parts that needs pose computation. MainIndex is required, Index2,...,IndexN are optional. If MainIndex is -1, command will compute all sub-regions' results If MainIndex is specific index, such as 5, and there are no other indexes, command will compute that specific sub-region(5) only If MainIndex, Index1, Index2, ..., IndexN are a serial of indexes(such as 1, 4, 7), command will only compute those given sub-regions(1,4,7).

Result String

When task runs successfully with StatusOnly as 0:

MGP/MGPA,<Status>,<Token>,<PartID>,<ResultCount>,<Status1>,<X_1>,<Y_1>,<T_1>[,<Status2>][,<X_2>][,<Y_2>][,<T_2>]...[,<StatusN>][,<X_n>][,<Y_n>][,<T_n>]CR/LF

Field	Arguments
<Status>	1: Task run successfully
<Token>	The Token identifying the corresponding command
<PartID>	PartID provided by command string
<ResultCount>	The count of requested sub-regions
<Status1>,[,<Status2>]...[,<StatusN>]	1: Pose computation runs successfully ErrorCode: Pose computation of given sub-region fails, by default ErrorCode = 0, but it can be other negative integers that defined by user in Custom Tool Block for feature finding
<X_1>,[,<X_2>]...[,<X_n>]	X component of target pose of requested sub-region

Field	Arguments
<Y_1>,[,<Y_2>]... [,<Y_n>]	Y component of target pose of requested sub-region
<T_1>,[,<T_2>]... [,<T_n>]	Theta component of target pose of requested sub-region

When task runs successfully with StatusOnly as 1:

MGP/MGPA,<Status>,<Token>,<PartID>,<ResultCount>,<Status1>[,<Status2>]...[,<StatusN>]CR/LF

When task fails:

MGP/MGPA,<Status>,<Token>,<PartID>CR/LF

Field	Arguments
<Status>	ErrorCode: Task fails
<Token>	The Token identifying the corresponding command
<PartID>	PartID provided by command string

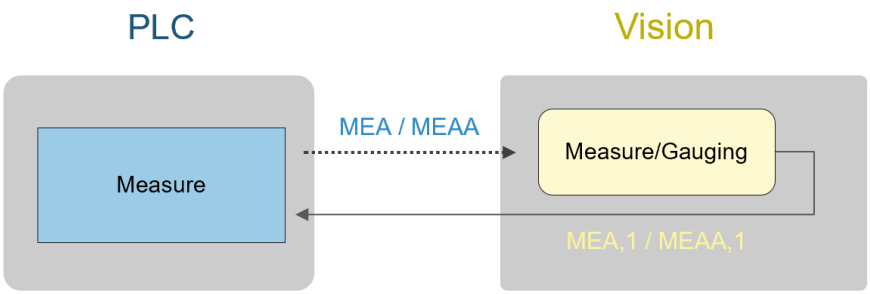
Note:
Before GP/GPA/MGP/MGPA command is sent, the external device needs to make sure all related LF/LFA commands have already finished execution. Otherwise the calculated result of x, y, theta by GP/GPA/MGP/MGPA command would be based on last time's run time feature result. In order to confirm that, the external device can compare the <Token> in the acknowledge signal with the <Token> in the received result string to see if they are the same after a LF/LFA command is sent out. If so, it means that the LF/LFA command has finished execution, otherwise, the feature finding task is still running.

Inspection Commands

Inspection command is used to run measurement or gauging on found features and output the results. It supports two commands: MEA and MEAA. MEA command will run the requested task synchronously, MEAA command on the other hand runs task asynchronously.

MEA/MEAA

Run measurement. This is the last command in a multi-step alignment sequence, and it assumes all features have been located successfully with preceding “LF” commands.



Command String

The two commands share the same command format. The only difference is the command key.

Command string for MEA command:

MEA,<EncodedID>,<PartID>,<RequestData>[,<UserString>]

Command string for MEAA command:

MEAA,<EncodedID>,<PartID>,<RequestData>[,<UserString>]

Field	Arguments
<EncodedID>	Alignment task + Process Tag(2000)
<PartID>	Serial number of part if it has, or any string if it is consistent for one part (PartID used here should be the same as used in LF commands if they are from the same part)
<RequestData>	0: return string only feedback OK/NG information 1: return string feedback OK/NG and measurement result values
[,<UserString>]	Optional, for user to customize some extra data

Result String

When task runs successfully:

MEA/MEAA,<Status>,<Token>,<PartID>,<ResultStatus>,<DataCount>,[<Data1>,][<Data1>,] ...[<DataN>,] CR/LF

Field	Arguments
<Status>	1: Task run successfully
<Token>	The Token identifying the corresponding command
<PartID>	PartID provided by command string
<ResultStatus>	0: Measurement results are out of specs 1: Measurement results are within specs
[<DataCount>]	The count of measurement results
[<Data1>,][<Data1>,] ... [<DataN>,]	Each measurement result data, optional

When task fails:

MEA/MEAA,<Status>,<Token>,<PartID>

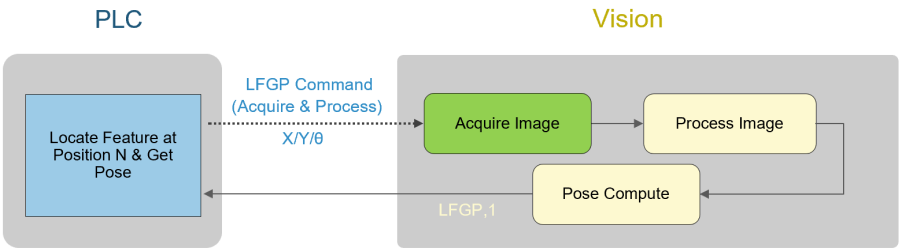
Field	Arguments
<Status>	ErrorCode: Task fails
<Token>	The Token identifying the corresponding command
<PartID>	PartID provided by command string

One Step Commands

For alignment or inspection applications in which the end of one feature finding task is surely followed by the beginning of an alignment task during run time, user can choose one-step command which merges the last LF command with the following alignment command (GP/MGP/MEA command) to trigger feature finding and alignment tasks at one time. These one-step commands are: LFGP, LFMGP, and LFMEA commands.

LFGP

For alignment application, run the feature locating at the last acquisition position in the requested feature finding task, and then run pose computation in the requested alignment task.



Command String

LFGP,<EncodedID0>,<EncodedID1>,<PartID>,<ResultMode>,<X>,<Y>,<Z>,<A>,<B>,<C>[,<UserString>]

Field	Arguments
<EncodedID0>	EncodedID0 = StepID for feature finder task's last position
<EncodedID1>	EncodedID1 = Alignment task + Process Tag(2000)
<PartID>	Serial number of part if it has, or any string if it is consistent for one part (PartID used here should be the same as used in LF commands if they are from the same part)
<ResultMode>	Abs (absolute value) or Off (offset from the current pose)
<X>	The X-coordinate of the current position of the motion system
<Y>	The Y-coordinate of the current position of the motion system
<Z>	0 (not used)
<A>	The theta(in degrees) of the current position of the motion system
<B>	0 (not used)
<C>	0 (not used)
[,<UserString>]	Optional, for user to customize some extra data

Result String

When task runs successfully:

LFGP,<Status>,<Token>,<ProductID>,<X>,<Y>,<Z>,<A>,<B>,<C>CR/LF

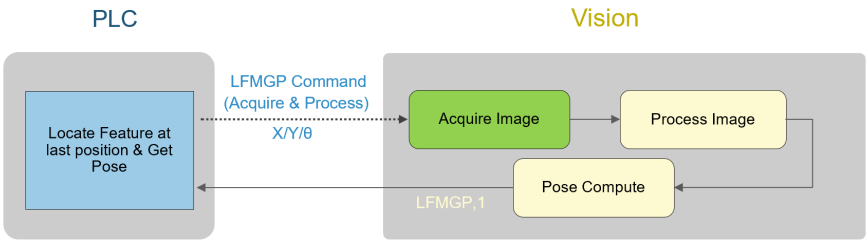
Field	Arguments
<Status>	1
<Token>	The Token identifying the corresponding command
<PartID>	The PartID provided by command string
<X>	The X-coordinate of the destination position for the motion system
<Y>	The Y-coordinate of the destination position for the motion system
<Z>	0 (not used)
<A>	The theta(in degrees) of the destination position for the motion system
<B>	0 (not used)
<C>	0 (not used)

When task fails:

Field	Arguments
<Status>	ErrorCode
<Token>	The Token identifying the corresponding command
<PartID>	PartID provided by command string

LFMGP

Runs multi-part feature locating at the last acquisition position of the feature finding task and pose computation in the alignment task sequentially.



Command string

LFMGP,<EncodedID0>,<EncodedID1>,<PartID>,<StatusOnly>,<ResultMode>,<X>,<Y>,<Z>,<A>,<B>,<C>,<UserString>,<MainIndex>[,<Index2>]...[,<IndexN>]

Field	Arguments
<EncodedID0>	EncodedID0 = StepID for feature finder task's last position
<EncodedID1>	Alignment task + Process Tag(2000)
<PartID>	Serial number of part if it has, or any string if it is consistent for one part (PartID used here should be the same used in LF commands if they are from the same part)
<StatusOnly>	0: return string feedback OK/NG information as well as target X,Y,Theta for each sub-region. 1: return string only feedback OK/NG for each sub-region
<ResultMode>	Abs (absolute value) or Off (offset from the current pose)
<X>	The X-coordinate of the current position of the motion system
<Y>	The Y-coordinate of the current position of the motion system
<Z>	0 (not used)
<A>	The theta(in degrees) of the current position of the motion system
<B>	0 (not used)
<C>	0 (not used)
,<UserString>	For user to customize some extra data(required field, not optional)
<MainIndex> [,<Index2>]... [,<IndexN>]	Indexes of sub-regions of parts that needs pose computation. MainIndex is required, Index2,...,IndexN are optional. If MainIndex is -1, command will compute all sub-regions' results If MainIndex is specific index, such as 5, and there is no other indexes, command will compute that specific sub-region(5) only If MainIndex, Index1, Index2, ..., IndexN are a serial of indexes(such as 1, 4, 7), command will only compute those given sub-regions(1,4,7).

Result String

When task runs successfully with StatusOnly set as 0 in command string:

LFMGP,<Status>,<Token>,<PartID>,<ResultCount>,<Status1>,<X_1>,<Y_1>,<T_1>[,<Status2>][,<X_2>][,<Y_2>][,<T_2>]...[,<StatusN>][,<X_n>][,<Y_n>][,<T_n>]CR/LF

Field	Arguments
<Status>	1: Task run sucessfully
<Token>	The Token identifying the corresponding command
<PartID>	PartID provided by command string
<ResultCount>	The count of requested sub-regions
<Status1>, [,<Status2>]... [,<StatusN>]	1: Pose computation runs successfully ErrorCode: Pose computation of given sub-region fails, by default ErrorCode = 0, but it can be other negative integers that defined by user in Custom Tool Block for feature finding

Field	Arguments
<X_1>,[,<X_2>]... [,<X_n>]	X component of target pose of requested sub-region
<Y_1>,[,<Y_2>]... [,<Y_n>]	Y component of target pose of requested sub-region
<T_1>,[,<T_2>]... [,<T_n>]	Theta component of target pose of requested sub-region

When task runs successfully with StatusOnly set as 1 in command string:

LFMGP,<Status>,<Token>,<PartID>,<ResultCount>,<Status1>[,<Status2>]...[,<StatusN>]CR/LF

The field arguments are the same as described above.

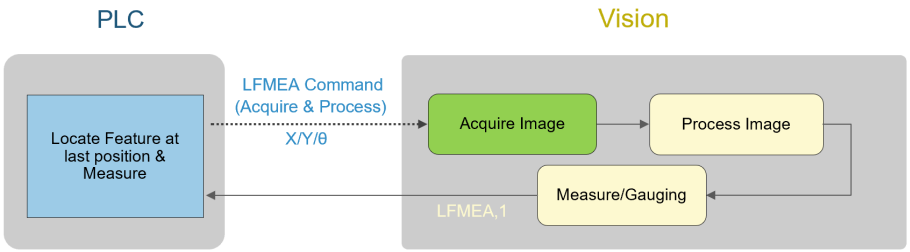
When task fails:

LFMGP,<Status>,<Token>,<PartID>

Field	Arguments
<Status>	ErrorCode: Task fails
<Token>	The Token identifying the corresponding command
<PartID>	PartID provided by command string

LFMEA

Runs the feature locating at the last acquisition position of the feature finding task and then inspection in the alignment task sequentially in synchronous mode.



Command String

LFMEA,<EncodedID0>,<EncodedID1>,<PartID>,<RequestData>,<X>,<Y>,<Z>,<A>,<B>,<C>[,<UserString>]

Field	Arguments
<EncodedID0>	EncodedID0 = StepID for feature finder task's last position
<EncodedID1>	EncodedID1 = Alignment task + Process Tag(2000)
<PartID>	Serial number of part if it has, or any string if it is consistent for one part (PartID used here should be the same as used in LF commands if they are from the same part)
<RequestData>	0: return string only feedback OK/NG information 1: return string feedback OK/NG and measurement result values
<X>	The X-coordinate of the current position of the motion system
<Y>	The Y-coordinate of the current position of the motion system
<Z>	0 (not used)
<A>	The theta(in degrees) of the current position of the motion system
<B>	0 (not used)
<C>	0 (not used)
[,<UserString>]	Optional, for user to customize some extra data

Result String

When task runs sucessfully:

LFMEA,<Status>,<Token>,<PartID>,<ResultStatus>[,<DataCount>] [,<Data1>] [,<Data2>]... [,<DataN>]CR/LF

Field	Arguments
<Status>	1
<Token>	The Token identifying the corresponding command
<PartID>	The PartID provided by command string
<ResultStatus>	0: Measurement results are out of specs 1: Measurement results are within specs
[,<DataCount>]	The quantity of measurement results
[,<Data1>] [,<Data2>]... [,<DataN>]	Each measurement result data.

When task fails:

LFMEA,<Status>,<Token>,<PartID>

Field	Arguments
<Status>	ErrorCode
<Token>	The Token identifying the corresponding command
<PartID>	PartID provided by command string

Command String Overview

Here is an overview of all commands in AlignPlus.

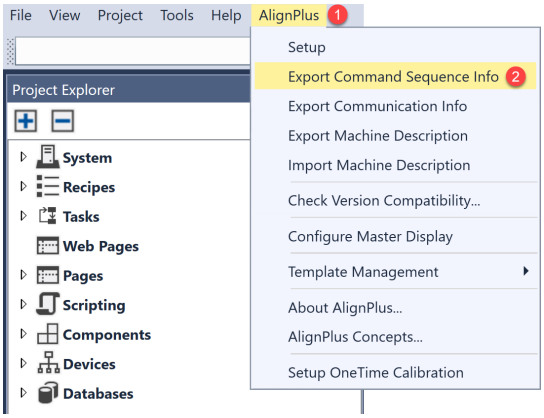
Command Type	Description	Command	Encode ID	Parameters	Return
Motion Guided Hand-eye Calibration	Hand-eye Calibration Start	HEB	,<EncodedID>, EncodedID: Calib<StepID> | Process only	/	HEB,<Status> CR/LF Status : 1 : Success. 0 : Fail
	Each Calibration Point	HE	,<EncodedID>, EncodedID: Calib<StepID> | Acquire & Process	<TargetID>,<X>,<Y>,<Z>,<A>,<B>,<C>	HE,<Status> CR/LF Status : 1 : Success. 0 : Fail
	Hand-eye Calibration End	HEE	,<EncodedID>, EncodedID: Calib<StepID> | Process only	/	HEE,<Status> CR/LF Status : 1 : Success. 0 : Fail
Vision Guided Hand-eye Calibration	Hand-eye Calibration Start	ACB	,<EncodedID>, EncodedID: Calib<StepID> | Process only	<TargetID>,<X>,<Y>,<Z>,<A>,<B>,<C>	ACB,<Status>, <X>,<Y>,<Z>,<A>,<B>,<C> CR/LF Status : 2 : Target was found, return next position 0 : Fail (Target position was not found.)
	Each Calibration Point	AC	,<EncodedID>, EncodedID: Calib<StepID> | Acquire & Process	<TargetID>,<X>,<Y>,<Z>,<A>,<B>,<C>	AC,<Status>, <X>,<Y>,<Z>,<A>,<B>,<C> CR/LF Status : 2 : Success, return the next position 1 : Success, calibration done. 0 : Fail
Cross Calibration	Calib at first position	IC	,<EncodedID>, EncodedID: CrossPos0<StepID> | Acquire only	<X>,<Y>,<Z>,<A>,<B>,<C>	IC,<Status>,< PartID > CR/LF Status : 1 : Success ; <= 0 : Fail
	Calib at second position	IC	,<EncodedID>, EncodedID: CrossPos1<StepID> | Acquire only	<X>,<Y>,<Z>,<A>,<B>,<C>	IC,<Status>,< PartID > CR/LF Status : 1 : Success ; <= 0 : Fail
	Calib at last position	IC	,<EncodedID>, EncodedID: CrossPos N<StepID> | Acquire & Process	<X>,<Y>,<Z>,<A>,<B>,<C>	IC,<Status>,< PartID > CR/LF Status : 1 : Success ; <= 0 : Fail
Training Pattern &Pose Golden Pose	Train pattern image at non-last position	TA	,<EncodedID>, EncodedID: Pos0<StepID> | Acquire only	/	TA,<Status> CR/LF Status : 1 : Success <= 0 : Fail
	Train pattern image at last position	TA	,<EncodedID>, EncodedID: Pos1<StepID> | Acquire & Process	/	TA,<Status> CR/LF Status : 1 : Success <= 0 : Fail

Command Type	Description	Command	Encode ID	Parameters	Return
Training Pattern &Pose AlignTo Gripper VGR	Train 1st pattern image	TT	,<EncodedID>, EncodedID: Pos0<StepID> | Acquire only	<X>,<Y>,<Z>,<A>,<B>,<C>	TT,<Status>CR/LF Status : 1 : Success <= 0 : Fail
	Train 2nd pattern image	TT	,<EncodedID>, EncodedID: Pos1<StepID> | Acquire & Process	<X>,<Y>,<Z>,<A>,<B>,<C>	TT,<Status>CR/LF Status : 1 : Success <= 0 : Fail
	Record golden pose of Robot	TTR	,<EncodedID>, EncodedID: Pos1<StepID> | Process	<X>,<Y>,<Z>,<A>,<B>,<C>	TTR,<Status>CR/LF Status : 1 : Success <= 0 : Fail
Alignment Sync Mode	Find Feature in Position1	LF	,<EncodedID>, EncodedID: Pos0<StepID> | Acquire only	<PartID>,<X>,<Y>,<Z>,<A>,<B>,<C> [,<UserString>]	LF,<Status>, <Token>,<PartID> CR/LF Status : 1 : Success <= 0 : Fail
	Find Feature in Position2	LF	,<EncodedID>, EncodedID: Pos1<StepID> | Acquire & Process	<PartID>,<X>,<Y>,<Z>,<A>,<B>,<C> [,<UserString>]	LF,<Status>, <Token>,<PartID> CR/LF Status : 1 : Success <= 0 : Fail
	Compute delta pose	GP	,<EncodedID>, EncodedID: Pose Computer <StepID> | Process only	<PartID>,<ResultMode>,<X>,<Y>,<Z>, <A>,<B>,<C>	GP,<Status> ,<Token>, PartID >, <X>,<Y>,<Z>,<A>,<B>,<C> CR/LF Status : 1 : Success <= 0 : Fail
	Compute poses for multi-part pickup	MGP	,<EncodedID>, EncodedID: Pose Computer <StepID> | Process only	<PartID>,<StatusOnly>,<ResultMode>, <X>,<Y>,<Z>, <A>,<B>,<C>,<UserString>,<MainIndex>[,<Index2>]...[,<IndexN>]	MGP,<Status>,<Token>,<PartID>,<ResultCount>, <Status1>,<X_1>,<Y_1>,<T_1> [,<Status2>][,<X_2>][,<Y_2>] [,<T_2>]...[,<StatusN>][,<X_n>] [,<Y_n>][,<T_n>]CR/LF Status : 1 : Success <= 0 : Fail
	Measure or gauge	MEA	,<EncodedID>, EncodedID: Inspection task <StepID> | Process only	<PartID>,<RequestData> [,<UserString>]	MEA,<Status>,<Token>,<PartID>,<ResultStatus>, <DataCount>,[<Data1>,] [<Data1>,] ...[<DataN>,] CR/LF Status : 1 : Success <= 0 : Fail

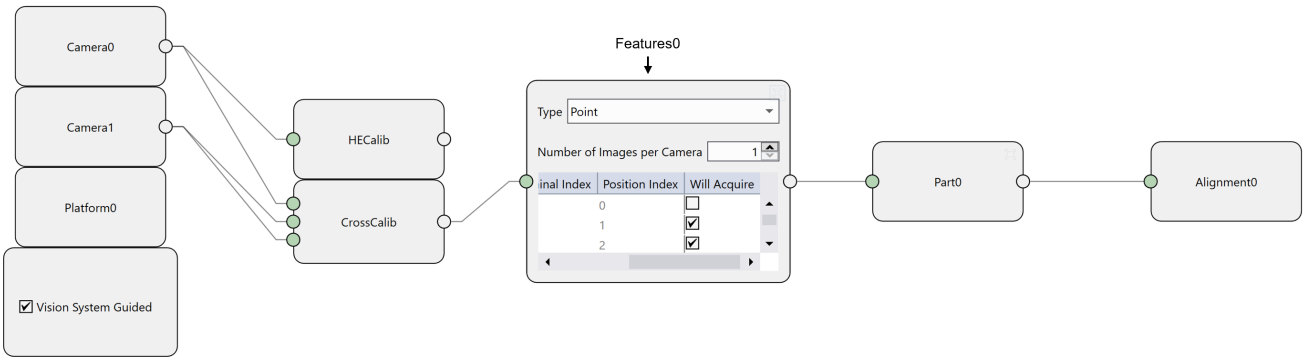
Command Type	Description	Command	Encode ID	Parameters	Return
Alignment Async Mode	Find Feature in Position1	LFA	,<EncodedID>, EncodedID: Pos0<StepID> | Acquire only	<PartID>,<X>,<Y>,<Z>,<A>,<B>,<C> [,<UserString>]	LFA,<Status>, <Token>,<PartID > CR/LF Status : 1 : Success <= 0 : Fail
	Find Feature in Position2	LFA	,<EncodedID>, EncodedID: Pos1<StepID> | Acquire & Process	<PartID>,<X>,<Y>,<Z>,<A>,<B>,<C> [,<UserString>]	LFA,<Status>, <Token>,<PartID > CR/LF Status : 1 : Success <= 0 : Fail
	Compute delta pose	GPA	,<EncodedID>, EncodedID: Pose Computer <StepID> | Process only	<PartID>,<ResultMode>,<X>,<Y>,<Z>, <A>,<B>,<C>	GP,<Status> ,<Token>, PartID >, <X>,<Y>,<Z>,<A>,<B>,<C> CR/LF Status : 1 : Success <= 0 : Fail
	Compute poses for multi-part pickup	MGPA	,<EncodedID>, EncodedID: Pose Computer <StepID> | Process only	<PartID>,<StatusOnly>,<ResultMode>, <X>,<Y>,<Z>, <A>,<B>,<C>,<UserString>,<MainIndex>[,<Index2>]...[,<IndexN>]	MGPA,<Status>,<Token>,<PartID>,<ResultCount>, <Status1>,<X_1>,<Y_1>,<T_1> [,<Status2>][,<X_2>][,<Y_2>] [,<T_2>]...[,<StatusN>][,<X_n>] [,<Y_n>][,<T_n>]CR/LF Status : 1 : Success <= 0 : Fail
	Measure or gauge	MEAA	,<EncodedID>, EncodedID: Inspection task <StepID> | Process only	<PartID>,<RequestData> [,<UserString>]	MEAA,<Status>,<Token>,<PartID>,<ResultStatus>, <DataCount>,[<Data1>,] [<Data1>,] ...[<DataN>,] CR/LF Status : 1 : Success <= 0 : Fail
Alignment One-step Command	Find Feature in Last Position and Compute delta pose for single part	LFGP	,<EncodedID0>,<EncodedID1>, EncodedID0: Find Feature last Position<StepID>| Acqure & Process EncodedID1: Pose Computer<StepID>| Process only	<PartID>,<ResultMode>,<X>,<Y>,<Z>, <A>,<B>,<C>[,<UserString>]	LFGP,<Status>,<Token>,<PartID >, <X>,<Y>,<Z>,<A>,<B>,<C> CR/LF Status : 1 : Success <= 0 : Fail
	Find Feature in Last Position and Compute delta pose for multiple parts pickup	LFMGP	,<EncodedID0>,<EncodedID1>, EncodedID0: Find Feature last Position<StepID>| Acqure & Process EncodedID1: Pose Computer<StepID>| Process only	LFMGP,<EncodedID1>,<ProductID2>,<StatusOnly>,<ResultMode>, <X>,<Y>,<Z>,<A>,<B>,<C>,<UserString>,<MainIndex>[,<Index2>]... [,<IndexN>]	LFMGP,<Status>,<Token>,<PartID>,<ResultCount>, <Status1>,<X_1>,<Y_1>,<T_1> [,<Status2>][,<X_2>][,<Y_2>] [,<T_2>]...[,<StatusN>][,<X_n>] [,<Y_n>][,<T_n>]CR/LF Status : 1 : Success <= 0 : Fail
	Find Feature in Last Position and run measurement/gauging	LFMEA	,<EncodedID0>,<EncodedID1>, EncodedID0: Find Feature last Position<StepID>| Acqure & Process EncodedID1: Inspection task<StepID>| Process only	<PartID>,<RequestData>,<X>,<Y>,<Z>, <A>,<B>,<C>[,<UserString>]	LFMEA,<Status>,<Token>,<PartID>,<ResultStatus> [,<DataCount>] [,<Data1>] [,<Data2>]... [,<DataN>]CR/LF Status : 1 : Success <= 0 : Fail

Export Command Strings

AlignPlus provides reference commands for the current application. This helps save user's time on command encoding. To export the reference commands, please click "Export Command Sequence Info" under "AlignPlus" menu and save the .txt file.



In the configuration example below, the task StepIDs are as follow:




```
StepID = 0                    HECalib_Loop
StepID = 1                    HECalib_ComputeCalibResults, Camera position: 0
StepID = 2                    CrossCalib, Camera position: 0
StepID = 3                    CrossCalib, Camera position: 1
StepID = 4                    CrossCalib, Camera position: 2
StepID = 5                    Features0, Calibration: CrossCalib, camera position: 1
StepID = 6                    Features0, Calibration: CrossCalib, camera position: 2
StepID = 7                    Alignment0
```

The exported commands are:

//Group Name:Calibrate

//HandEye calibration

```
HEB,2001
HE,1,0,x,y,0,theta,0,0
HEE,2001
```

//Auto calibration

```
ACB,2001
```

```
AC,1,0,x,y,0,theta,0,0
```

//cross calibration at position 0

```
IC,1002,0,0,0,0,0,0
```

//cross calibration at position 1

```
IC,1003,0,0,0,0,0,0
```

//cross calibration at position 2

```
IC,4,0,0,0,0,0,0
```

//Group Name :Alignment0

//Feature golden pose training at position 1

```
TA,1005
```

//Locate features at position 1

```
LF,1005,productID,0,0,0,0,0,0,userString
```

//Feature golden pose training at position 2

```
TA,6
```

//Locating feature and computing target pose

```
LFGP,6,2007,productID,resultMode[Abs/Off],x,y,0,theta,0,0,userString
```

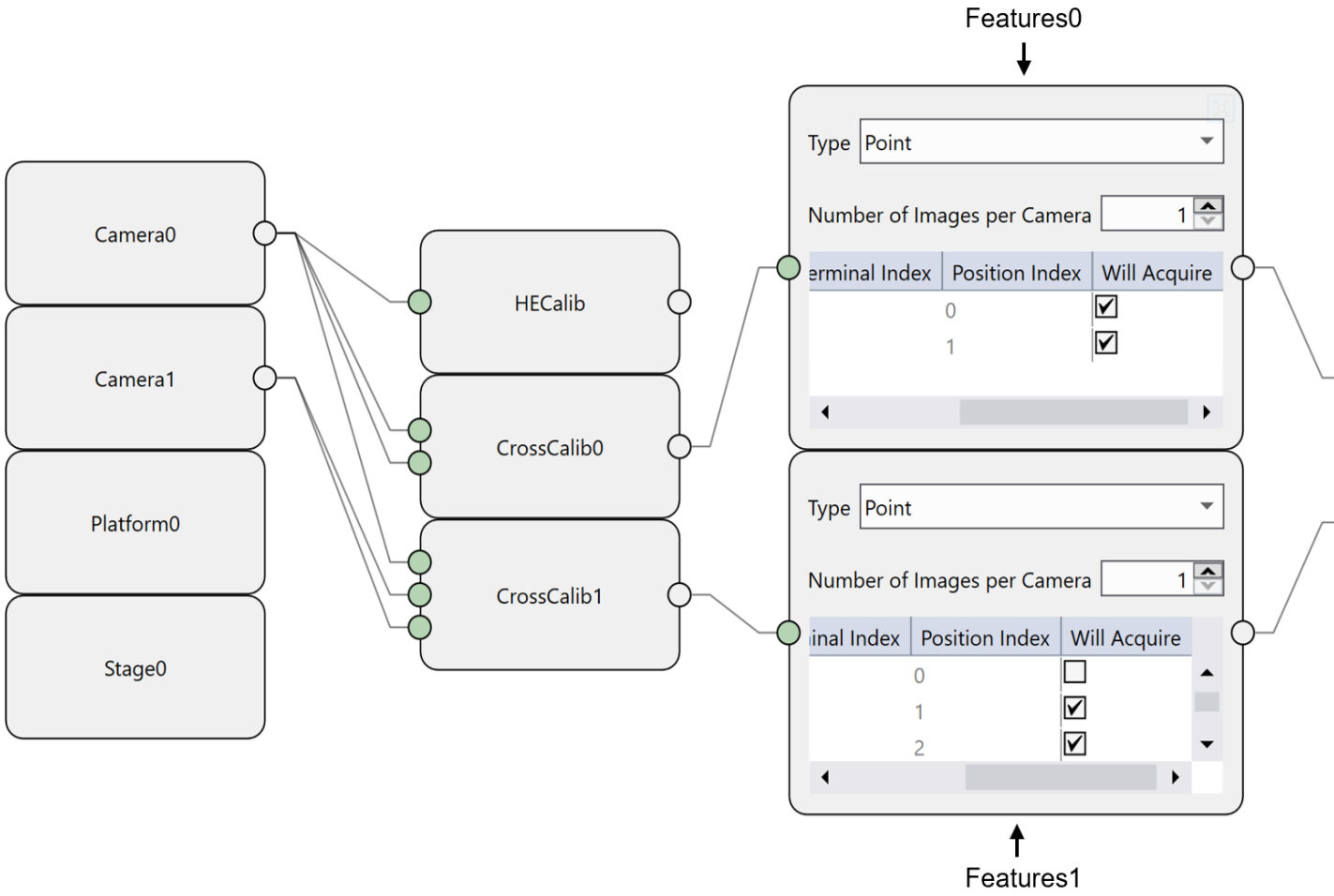
//Computing target pose

```
GPA,2007,resultMode[Abs/Off],x,y,0,theta,0,0,userString
```

Note:
<X>, <Y>, <Theta> in IC, LF and LFGP commands should be replaced with real motion's current x, y, theta values.

The LF command at position 2 are merged with GP command as one LFGP command in reference commands, however, they could also be split into two commands and runs separately.

Here is an assembly guided pick application:



Here are the StepIDs for each task defined in CommandHandler:

```
StepID = 0                    HECalib_Loop
StepID = 1                    HECalib_ComputeCalibResults, Camera position: 0
StepID = 2                    CrossCalib0, Camera position: 0
StepID = 3                    CrossCalib0, Camera position: 1
StepID = 4                    CrossCalib1, Camera position: 0
StepID = 5                    CrossCalib1, Camera position: 1
StepID = 6                    CrossCalib1, Camera position: 2
StepID = 7                    Features0, Calibration: CrossCalib0, camera position: 0
StepID = 8                    Features0, Calibration: CrossCalib0, camera position: 1
StepID = 9                    Features1, Calibration: CrossCalib1, camera position: 1
StepID = 10                    Features1, Calibration: CrossCalib1, camera position: 2
StepID = 11                    Alignment0
StepID = 12                    HECalib_MotionAnalysis
StepID = 13                    HECalib_MotionAnalysisPoseGenerator, Camera position: 0
```

The exported commands are:

//Group Name :Calibrate

//HandEye calibration

```
HEB,2001
HE,1,0,x,y,0,theta,0,0
HEE,2001
```

```
//Auto calibration
ACB,2001
AC,1,0,x,y,0,theta,0,0

//cross calibration at position 0
IC,1002,0,0,0,0,0,0
//cross calibration at position 1
IC,3,0,0,0,0,0,0

//cross calibration at position 0
IC,1004,0,0,0,0,0,0
//cross calibration at position 1
IC,1005,0,0,0,0,0,0
//cross calibration at position 2
IC,6,0,0,0,0,0,0

//Group Name :Alignment0
//Features0
//Feature golden pose training at position 0
TA,1007
//Locate features at position 0
LF,1007,productID,0,0,0,0,0,0,userString
//Feature golden pose training at position 1
TA,8
//Locate features at position 1
LF,8,productID,0,0,0,0,0,0,userString

//Features1
//Feature golden pose training at position 1
TA,1009
//Locate features at position 1
LF,1009,productID,0,0,0,0,0,0,userString
//Feature golden pose training at position 2
TA,10
//Motion golden pose training
TTR,10,x,y,0,theta,0,0
//Locating feature and computing target pose
```

```
LFGP,10,2011,productID,resultMode[Abs/Off],x,y,0,theta,0,0,userString
```

//Computing target pose

```
GPA,2011,resultMode[Abs/Off],x,y,0,theta,0,0,userString
```