Pally URCap

User Manual

Version 3.1

Table of Contents

1 - Document Revision Information

Date	Description	Author
15.11.2018	Initial draft	CM,LÅ,AL
08.02.2019	Checklist for new robot setup	CM
12.02.2019	Functional description, typical setup examples	CM
25.02.2019	Comments resolved, screenshots updated	CM
24.05.2019	Lifting column, shim paper, updated screenshots	CM
29.08.2019	Gripper and TCP configuration rewritten	CM
04.10.2019	Added gripper optimization, grippers with offset	CM
19.10.2019	Adding description of ‘Partial Pallet’-feature	ME
27.03.2020	Installation process with new GUI	ME
18.05.2020	Description of json format, new functions in 2.5.0	CM
26.05.2020	New Advanced tab in installation node	ME
28.09.2020	New functions in 2.6.0	CM
11.01.2021	Updates related to 2.6.1	CM
29.01.2021	Dual product mode updated for v2.7.0	ME
26.04.2021	Updates related to 2.7.1, new illustrations	CM
14.06.2021	Updates related to 2.8.0	CM
01.09.2021	Explanation of Smart Exit parameters, v2.8.1	CM
25.10.2021	v2.9.0, new gripper.json format, automatic single pick, updated screenshots.	CM
22.12.2021	Vention MachineMotion V2 added	CM
28.01.2022	v2.10, operator panel, UR+ toolbar, calibration test wizard, automated pattern selection added	CM
31.03.2022	v2.11, added Piab grippers and Portuguese language	CM
28.06.2022	v2.12, Demo Mode, Instant License, compatibility notes added	CM

20.10.2022	v2.13 Order Management	CM
21.11.2022	V3.0 related changes	CM
31.01.2023	Added description of alternative reach check option	CM
05.05.2023	v3.1, Easy path planning, Finish pallet buttons	CM

2 - License

Disclaimer

The information contained herein is the property of Rocketfarm AS and shall not be reproduced in whole or in part without prior written approval of Rocketfarm AS. The information herein is subject to change without notice and should not be construed as a commitment by Rockerfarm AS. Rocketfarm AS assumes no responsibility for any errors or omissions in this document.

The Pally logo is a registered trademark of Rocketfarm AS.

3 - Requirements

3.1 - Software Requirements

URCap API version 1.12 or higher is required.
CB-series: Polyscope version 3.15 or higher is required.
E-series: Polyscope version 5.10 or higher* is required.
* A separate installer package will be provided for Polyscope 6 series.

3.2 - Hardware Requirements

Due to size limitations, the palletizer URCap requires a UR10 robot.
On CB-series robots, only CB-series version 3.1 is fully supported.
CB 3.0 robots may be used with some limitations, see Compatibility with CB 3.0 robots

3.2.1 - Supported external hardware

Lifting columns

- Ewellix LiftKit
- Vention MachineMotion V1 and V2
- Linak Elevate
- Custom lifting column with 24V digital Up/Down signal

Grippers

- UniGripper CoLight Regular
- Schmalz FXCB with Foam or Suction cups
- Piab CPT series 300, 400, 500 mm with Foam or Suction cups
- Custom vacuum gripper with 24V digital signal
- Custom gripper
- Grippers with one or more individually controllable grip areas

3.3 - Compatibility notes

!	**Please read the following compatibility notes carefully before upgrading an existing Pally installation in production. Some changes in the recent versions may require new customer acceptance tests and/or modifications in the main robot program. Always make a full backup before upgrading.**

3.3.1 - Compatibility notes for version 2.6 and above

3.3.1.1 - Path planning

The path planning algorithm has significantly changed in version 2.6. It is recommended to test existing projects with full pallets of all products after upgrading to version 2.6.

3.3.1.2 - ProductCount

Since version 2.6 the "ProductCount" variable is now properly initialized to 0. In earlier versions it was initialized to -1 incorrectly. It is necessary to update custom code in callbacks that rely on the ProductCount variable.

3.3.2 - Compatibility notes for version 2.7.1 and above

3.3.2.1 - Pallet confirmation variables

From version 2.7.1 the pallet confirmation variables "rf_PS1_ok" and "rf_PS2_ok" should be updated with the following script command:
`palletmanager_daemon.write_key_value_static(name, value)`
The old method `palletmanager_daemon.write_key_value(0, name, value)` has no longer effect on the pallet confirmation state.
Programs that directly update the pallet confirmation variables should be verified.

3.3.3 - Compatibility notes for version 2.8 and above

3.3.3.1 - Execution of beforeZone and afterZone callbacks

From version 2.8.0 the beforeZone callback will be invoked when the lifting column has already reached the specified position but palletizing has not yet started. Similarly, the afterZone callback will be invoked after palletizing the last box has finished but the lifting column is still in the position as specified by the current zone. Programs that use a lifting column and have user code in beforeZone and afterZone callbacks should be verified.

3.3.3.2 - Execution of beforeGrab callback

From version 2.8.0 the pipeline architecture for all path planning calculations has been redesigned such that all calculations are already done when the program enters the beforeGrab callback. Changing some path-planning related internal rf_-variables in beforeGrab for smart exit or collision radius calculations will no longer have any effect on the next movement. Programs that modify internal (rf_) path planning variables directly in beforeGrab should be verified.

3.3.3.3 - Waiting time at pickup and release positions

From version 2.8.0 the waiting time at pickup and release positions has been optimized, in order to minimize the occurrence of protective stops caused by too early or too late change in the payload and center of gravity. The waiting time may become shorter or longer than in previous versions. Projects that are sensitive to the waiting times at pickup and release positions should be verified. For further information, see the new variables rf_joint_deviation_max and rf_joint_deviation_err under 9.4.6 - Other/Special

3.3.4 - Compatibility notes for version 2.9 and above

3.3.4.1 - URCap API 1.12

From version 2.9.0 the minimum required URCap API version is 1.12. which may require the robot to be upgraded to a significantly newer version of Polyscope. Always make a full backup before upgrading.

3.3.4.2 - Definition of multi-zone grippers (gripper.json)

From version 2.9.0 the structure of gripper.json has been changed. The zone positions are now defined in accordance with the tool coordinate system, which means the X and Y directions are inverted. This applies to all new gripper.json files that have the new format with the "properties" structure included, otherwise the old behavior is kept for compatibility reasons.

3.3.4.3 - Automatic multi-pick

From version 2.9.0 the palletizer program evaluates the maxGripAuto field in the pattern JSON file prior to maxGrip. When maxGripAuto is set to true, the program will ignore the maxGrip value and use the product and gripper dimensions and the number of installed product sensors to determine the maximum number of products to be picked in one turn. Existing projects with multi-pick should take this into account when creating new patterns.

3.3.4.4 - Multi-pick fewer boxes

From version 2.9.0 the optimizer will try to pick fewer boxes when the specified (maximum) amount of boxes cannot be palletized for some reasons, including out of reach issues and too small gripper size. In contrast to older versions that failed immediately with "This position cannot be palletized" the program will now retry with fewer boxes. This may introduce unpredictable behavior when the calibration of an existing system is modified.

3.3.4.5 - Optional posData in zones.json

From version 2.9.0 the "posData" values in zones.json are made optional. This makes it possible to use zones in combination with lifting column dynamic positioning.

3.3.4.6 - New callback onNextTask

From version 2.9.0 a new callback "onNextTask" will be automatically generated into the existing program tree when the Pally program node is opened in the Program Robot menu for the first time. The program has to be saved in order to keep the new structure.

3.3.5 - Compatibility notes for version 2.10 and above

3.3.5.1 - Pally Operator Panel (POP)

Functions of the Pally Operator Panel, also known as the Runtime GUI URCap, have been integrated into the Pally URCap. Installing two separate URCaps is no longer required. Compatibility with the older versions of POP are not maintained. Users are advised to uninstall all older versions of POP as they may not function properly any more.

3.3.5.2 - Maximum allowed payload with multi-pick

From version 2.10.0 the program will check the maximum allowed payload for the specific robot model when selecting more than one box for multi-pick. As a result, the program may reject picking multiple packages, or reduce the number of boxes being picked, in projects where the maximum payload could be previously exceeded.
Maximum allowed payload is derived from the serial number format.

3.3.6 - Compatibility notes for version 2.12.0 and above

3.3.6.1 - Ewellix LIFTKIT URCap compatibility

Some compatibility issues with Ewellix LIFTKIT URCap version 1.2.0 have been resolved. Users having Ewellix LIFTKIT URCap version 1.1.0 may receive the following false alarm "**Pillar not in position - press Continue to retry**" after pausing and resuming the robot while operating the lifting column. Users are advised to upgrade LIFTKIT to 1.2.0 or newer - provided by Ewellix.

3.3.6.2 - Lifting column Dynamic Positioning

The lifting column optimizer behavior has been significantly changed in version 2.12.0. Calculated lifting column positions may be different from in previous versions. Existing projects that use Dynamic Positioning should be verified. Projects that directly modify the `rf_lift_alt_boost` and `rf_lift_safe_down` parameters are advised to set `rf_lift_alt_boost_auto=False` and verify the program.

3.3.7 - Compatibility notes for version 3.0 and above

3.3.7.1 - Offline waypoint storage

Waypoints for all box positions are stored in a database, and the robot will repeat the same waypoints for each pallet as long as the calibration parameters remain unchanged. Changing path planning parameters will have no effect on the waypoints until the database is deleted. Changing calibration points, TCP, gripper, and lifting column parameters will mark the waypoints useless, which requires testing the pallet again.

3.3.7.2 - Service and Support plan

The program will lock access to some important settings when a Service and Support plan expires.

3.3.7.3 - URCap version in the active program

The program will store information about the URCap version last used when the program was saved. Upgrading to a newer URCap without a valid Service and Support plan will not work.

3.3.7.4 - Grip Quality

The formula for computing the "Grip Quality" value has been redesigned. It is highly recommended to test all patterns after upgrading from URCap version 2.x to 3.x.

3.3.7.5 - Pallet Manager Daemon variable name and visibility

The global variable `palletmanager_daemon` has been renamed to `Pally_daemon`. The legacy variable `palletmanager_daemon` is still available but only inside the Pally program node (initial MoveJ and callbacks).

3.4 - Compatibility with CB 3.0 robots

Due to hardware limitations in the CB 3.0 control boxes, the Pally URCap may not run as smoothly as expected. Consider the following steps before installing Pally URCap on these robots.

3.4.1 - Memory usage limitations on CB 3.0

Out of memory exceptions may occur during installation and configuration of the Pally URCap unless the following modification is made in the operating system.

- Find the file run_gui.sh under /root
- Find the line `RUN_GUI="java ...`
- Add the following option: `-XX:MaxPermSize=128m`

After the modifications, the line should look like this:

```
RUN_GUI="java -XX:MaxPermSize=128m
-Djava.library.path=/root/GUI/lib -jar bin/*.jar"
```

Note: Make sure all whitespaces are inserted properly as shown above.

Note: Make sure all capital letters and small letters are properly written. The operating system is case sensitive and requires the parameter names to be written exactly as shown above.

Note: Repeat the above steps every time after upgrading Polyscope.

If you are not familiar with operating system parameter tuning, follow the steps below:

- Connect a keyboard to the Teach Pendant or an USB connector in the control box
- Press Control+Alt+F1 to enter the console
- Log in with username `root` and password `easybot`
- Go to the /root folder by entering `cd /root`
- Type `nano run_gui.sh` to edit the file
- Make the above described modifications, then press Ctrl+X to save the changes
- Press Ctrl+Alt+F7 to return to the Polyscope screen
- Disconnect the keyboard
- Restart the robot

3.4.2 - Computational limitations on CB 3.0

In order to reduce calculation times and avoid lagging, keep the following rules in mind when creating pallet patterns.

- Use "lock direction" for all boxes wherever possible.
- Disable 2-ways and 4-ways gripper optimization and only add the enforced gripper orientation at positions where it is needed.
- Use the "inverse approach" option for all layers except the highest layers.
- Disable dynamic positioning of the lifting column, and control the lifting column through zones.json file(s)
- Reduce the Pally log level to Error or Warning

3.5 - Compatibility with other URCaps

The Pally URCap supports a wide range of hardware components, however, unexpected compatibility issues can occur with specific setups.

We have received feedback from customers using hardware controlled by additional URCap(s) that the robot program becomes unstable and stops randomly with the error

message “Runtime too much behind” on E-series and “Communication with controller lost” on CB-series robots. These error messages are related to insufficient CPU resources or high peaks in CPU usage: while both Pally and the third party URCap can work as standalone, they can’t be used together without further adjustments.

There are several options to control CPU usage of the Pally URCap. By default, the program utilizes as much CPU as possible, and provides fast calculations and low response times. In combination with another URCap with high CPU demand however, this can cause the program to stop. In this case we recommend the following actions:

- Reduce the CPU usage of the other URCap(s) if possible
- Change the Pally CPU optimizer parameter rf_cpuidle_ticks
    - On E-series, try one of the following values: rf_cpuidle_ticks=50, 20, or 1
    - On CB-series, try rf_cpuidle_ticks=5000, 2000, or 1
    - larger values correspond to faster calculations,
    - smaller values correspond to less peaks in CPU usage
- Reduce complexity of the pattern and path planning as described in the previous section Computational limitations on CB 3.0

4 - Functional description

4.1 - The palletizer workflow

The typical use case for a palletizing robot is described as follows:

- The operator starts the robot and the palletizer software,
- The operator enters the product name to be palletized,
- The robot starts palletizing on an empty pallet,
- When the pallet is full, the operator must replace it with an empty pallet
- Palletizing continues until the operator stops the program.

In addition to the above described default use case, the following scenarios are also available in Pally:

- Finish the current pallet immediately and begin a new empty pallet,
- Build a partial pallet, i.e. palletize fewer boxes and/or layers than usual,
- Continue an incomplete pallet, i.e. specify the starting layer/box position,
- Palletize from 2 pickup positions,
- Palletize 2 different products from 2 pickup positions on 2 pallets simultaneously.

Note: Every pallet is built of one specific product type. Thus, it is not possible to palletize different products on the same pallet.

4.2 - Pallet patterns

Every product has its own pallet layout, depending on the box dimensions and the label position on the boxes. Some products have to be palletized using interlocking layers, which means that every second layer is mirrored or rotated in order to improve pallet stability. Other products have to be palletized so that the boxes are put exactly on the top of each other. The palletizer software can handle these requirements by defining the pallet patterns.

Note: A pallet pattern is the logical representation of a pallets layout, which can be dynamically loaded into the program. The number of supported patterns is only limited by the available disk space.

4.3 - Pallet completion state

While pallet patterns define the expected layout of the pallet as a static model, the pallet completion state follows the actual status of the pallet currently being stacked. The pallet completion state keeps track of which exact box positions are already done, and which are waiting to be filled. The completion state is updated every time the robot has successfully moved a box from the pickup position and released it at the target position on the pallet.

Note: The robot will not be aware of any boxes that are moved to, on or from the pallet by the operator.

4.4 - Pickup position

The pickup position is at the end of the conveyor belt, where the boxes stop and wait until the robot picks them up.

Note: The pickup position must be stationary, i. e. the conveyor should be fixed to the floor.

One side of the conveyor should have a fixed side-guide, which the boxes can be aligned to. The other side-guide should be adjustable for different product sizes where needed, see **figure 1**. Depending on the box weight and dimensions, the robot can lift up multiple boxes at the same time. When multi-grip is available, the robot will align the gripper equally between the boxes to maximize gripping stability, as illustrated in **figure 2**.

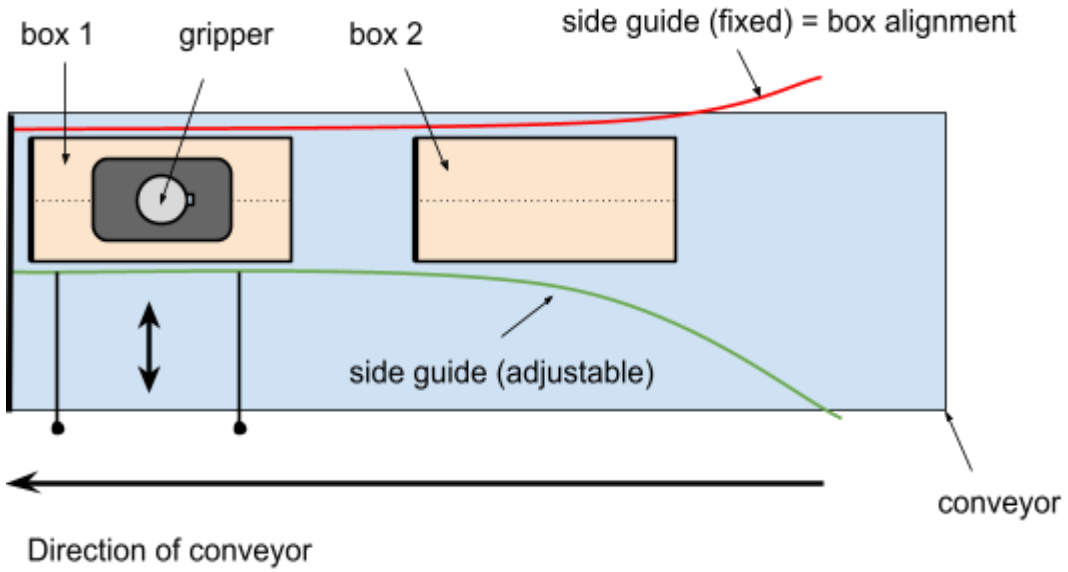


Figure 1: Pickup position with one product.

4.4.1 - Single-pick and multi-pick

Depending on the actual pallet pattern and gripper dimensions, it is possible to pick one or more boxes at the pickup position. To pick multiple boxes, the positions in the pattern file must follow each other by product length. The pick and place positions are automatically calculated for the given number of boxes.

The program will automatically retry with fewer boxes when the specified number of boxes cannot be palletized:

- when the gripper is too small to pick all boxes
- the pick position is out of reach
- the target position is out of reach
- collision-free path planning is not possible

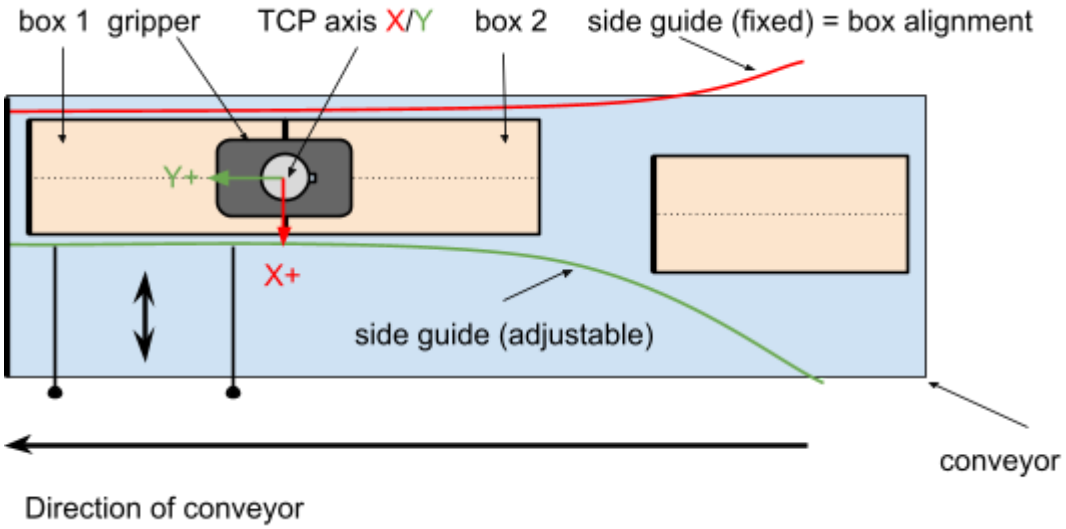


Figure 2: Pickup position with two products. Notice that the tool coordinate system is aligned to the conveyor direction.

4.4.2 - Gripper alignment at the pickup position - offset grip

To avoid accidental multi-picking, the front edge of the gripper is always aligned with the rear edge of the (last) product being picked. This is especially important when boxes are smaller than the gripper. In this case the gripper is not aligned centered above the box, which is a common misunderstanding when using Pally for the first time.

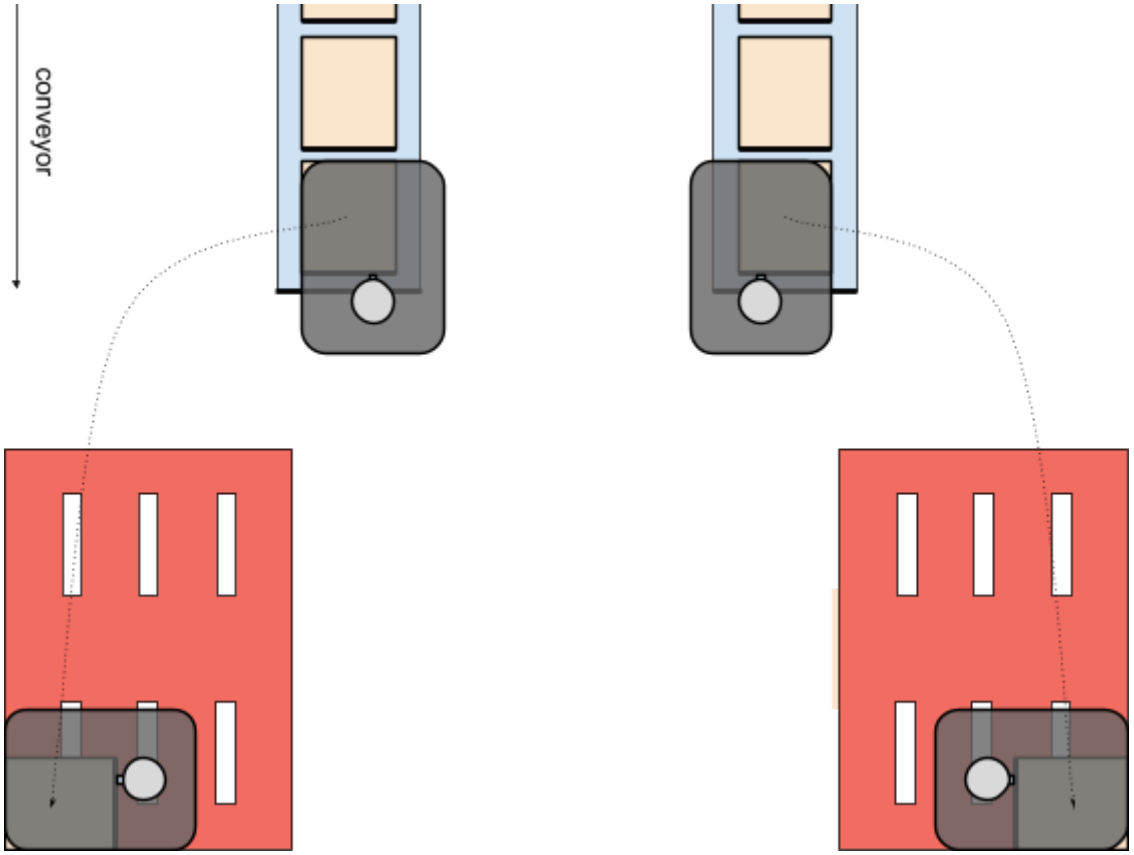


Figure 3: edge alignment optimization for large grippers

The first time calibration requires the gripper center aligned to the box center, as shown in the following diagram. In contrast to the calibration process, the real pick position will be calculated with respect to the edge alignment (when needed).

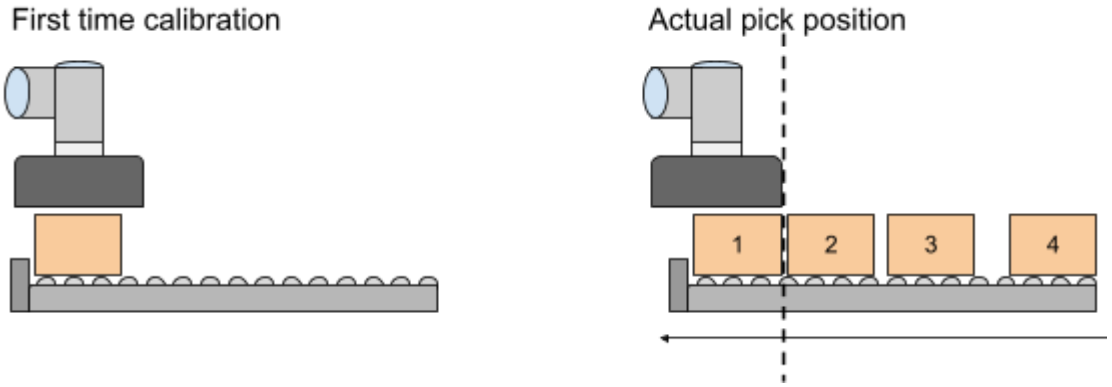


Figure 4: gripper alignment at pickup

4.4.3 - Gripper rotation at the pickup position - gripper optimization

In order to get better reach or performance, the program can automatically recalculate and use the gripper rotated by +90, -90, 180 degrees at the pickup position. This may be

necessary to reach the pallet corners or avoid collision with the base frame. The allowed rotations can be configured.

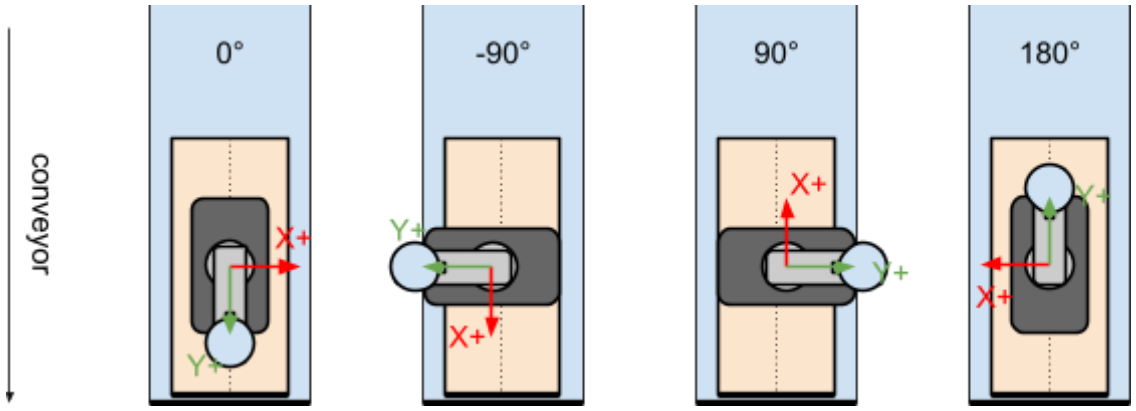


Figure 5: gripper rotation at pickup

There are 3 default optimization modes to choose from: 4-ways, 2-ways, and no optimization, which are listed below.

None	only 0°
2-ways	0° and 180°
4-ways	0°, -90°, 90°, and 180°

The optimizer combines gripper rotation and gripper alignment in order to maximize reach on the pallet. Which edges of the (possibly rotated) gripper are aligned to which edges of the box(es) depend on the actual target rotation on the pallet.

4.4.4 - Grip quality estimation

Based on the actual product and gripper dimensions, payload weight, and the number of boxes being picked, the program estimates a reference value called 'grip quality' which is used by the optimizer to choose between possible gripper rotations and reduce the number of boxes to be picked in one robot cycle.

4.4.5 - Smart acceleration

When the estimated grip quality value drops below a minimum threshold, the program moves the boxes with reduced acceleration in order to minimize the risk of dropping boxes during transport. The parameters of this algorithm can be configured, see 6.2.3 - Movement.

The program will try to choose a gripper rotation that has higher grip quality value. If the grip quality is still low, the program will reduce accelerations to avoid lost boxes. The grip quality minimum threshold is project and product specific, and provided as a tuning parameter.

4.5 - Path planning

The path planning algorithm is an essential part of the palletizer software. It calculates the optimal movement from the pickup position to the pallet position. These calculations are performed dynamically when the robot moves one or more boxes from the pickup to the pallet position.
After picking the products, the robot moves vertically up until the box is removed from between the side guides. Depending on the current pickup and target positions, this is followed by one or more waypoints in order to move and rotate the box(es) without collision to the robot base or the existing boxes on the pallet. The approach position is the last waypoint in the proximity of the final target position. The robot uses lower acceleration and speed from the approach to the final target position to improve the pallet accuracy.

Note: To ensure a smooth palletizer experience, some properties of the production line should be carefully measured.

4.5.1 - Above pickup position

The path starts with a position above the pickup point, to ensure the gripper moves vertically down against the box surface. At this point the program can automatically activate compressed air (if available) to clean the gripper foam and the box surface before picking.

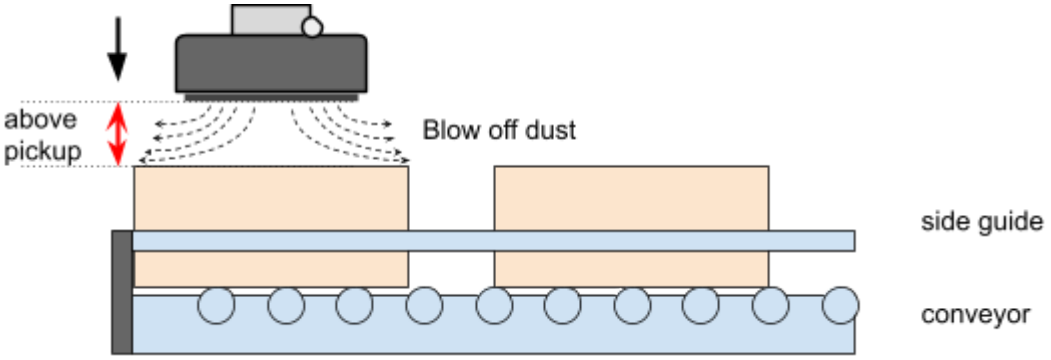


Figure 6 : gripper approaching box vertically from above

4.5.2 - The path planning algorithm

The purpose of the path planning algorithm is to move the box from the pick position to the pallet position without collisions and as fast as possible.
The path planning result depends on several settings that are introduced below.

First, the path planning algorithm will try to move the box on a linear path whenever possible, see **figure 7** and **figure 8**. After the box is taken from the pickup position, the algorithm tries to find one waypoint - called *smart exit* position - from where the shortest, direct linear movement is possible.

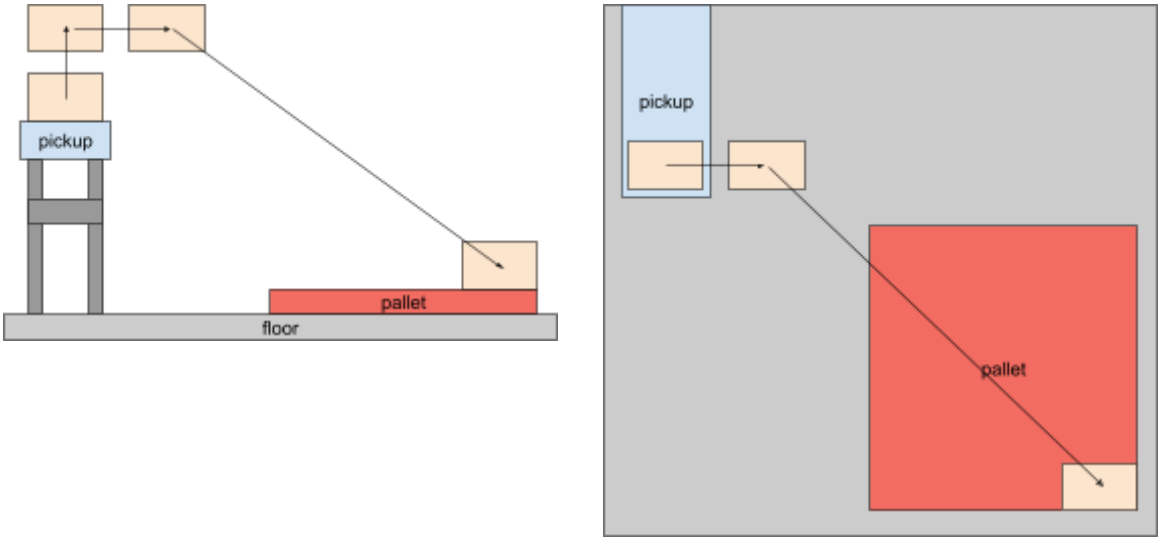


Figure 7: The primary path planning algorithm on lower layers.

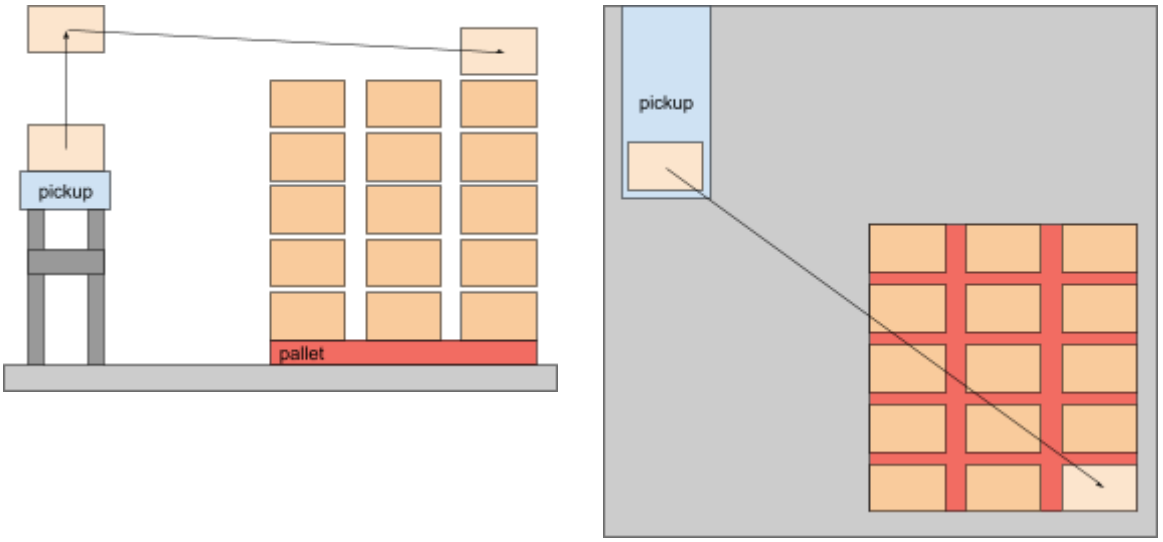


Figure 8: The primary path planning algorithm on higher layers.

It is possible to define a rectangular area where the path planning algorithm can search for an intermediate waypoint (smart exit) between box-free and approach position, to perform linear movements without collision to the robot base, see the green areas on **figure 9**.

Figure 9: Illustration of the *smart exit* search area.

When it is not possible to find a *smart exit* position from where a direct linear movement is possible, one or more waypoints are being inserted. In this case the program evaluates the pallet completion state (position of the already palletized boxes) and finds a collision-free path to the target position. It is often necessary to move the box above all other boxes, and then lower vertically to the target position, see **figure 10**. This path is normally longer than a direct linear movement, and hence more time demanding.

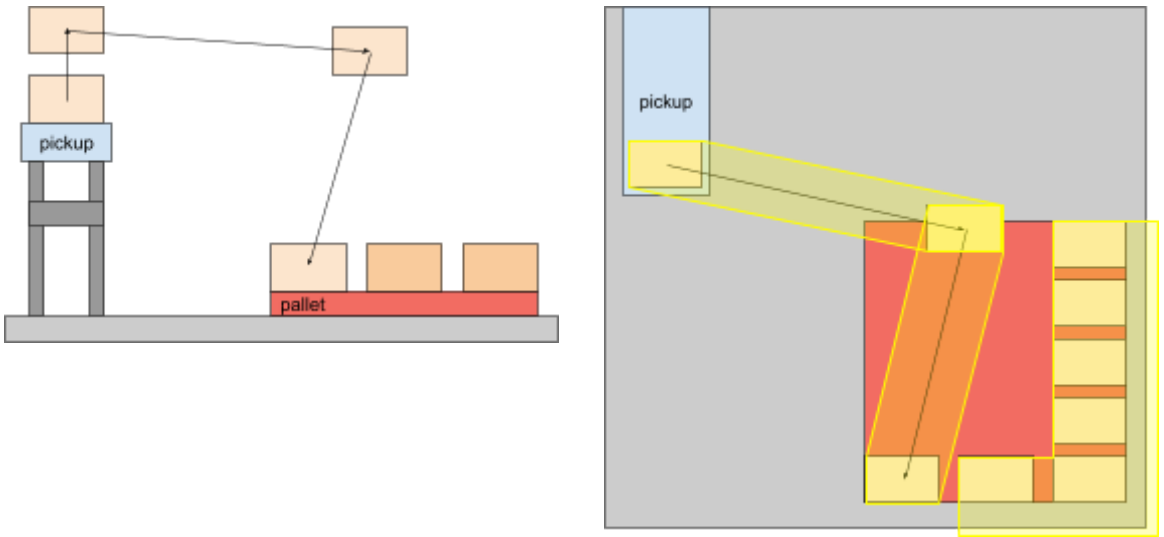


Figure 10: Path planning with an extra waypoint. Boxes on the same layer define a collision area to be avoided.

When the target position is below the conveyor level, it is often necessary to keep the box as high as possible until the robot leaves the conveyor area. This behavior can be configured in via the *high approach boost* settings: lower values let the box move closer to the top of the other boxes on the pallet, higher values keep the box closer to the *box free* height (**figure 11**).

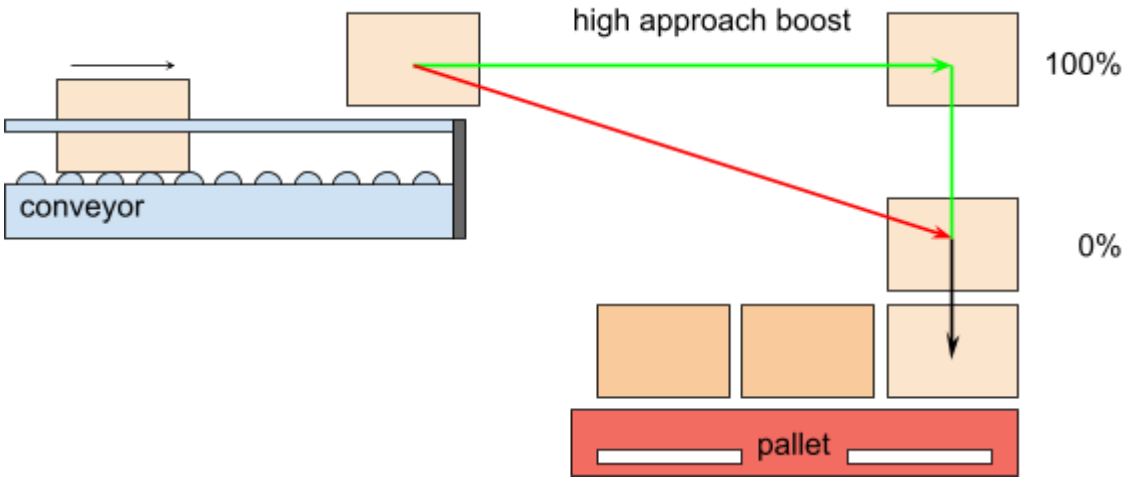


Figure 11: Illustration of the *High approach boost* parameters.

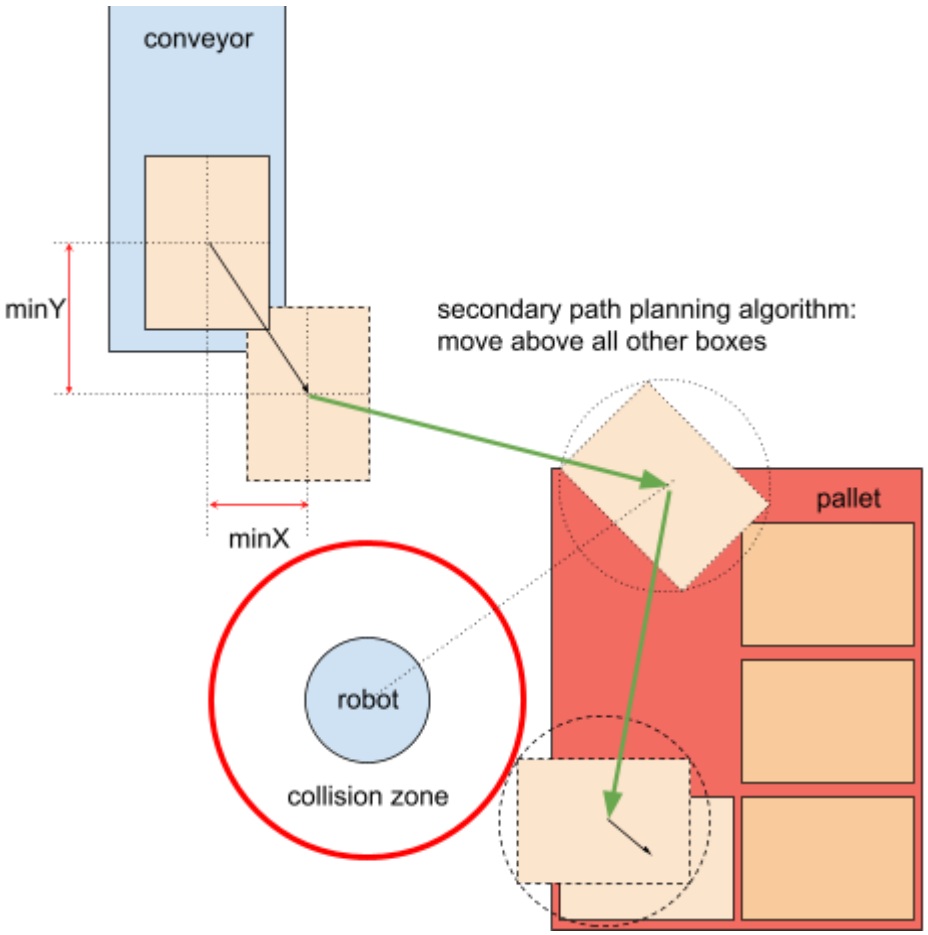


Figure 12: Illustration of when *High approach* is used.

4.5.3 - Approaching the target position

In order to improve precision, the last section of the movement is being performed with lower speed and acceleration.
Normally the robot moves the box into its final position by using the "Approach" distance, which is shown on **figure 13**.

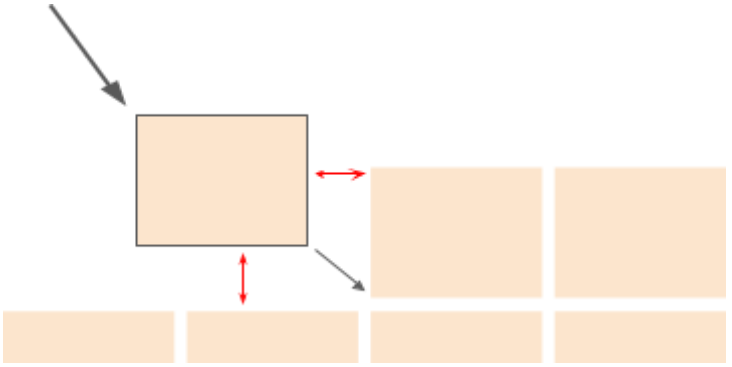


Figure 13: Approaching the target position with lower speed and acceleration

---

Note: To keep the necessary distance from the existing boxes on the pallet, the approach position may be recalculated dynamically if rotation occurs between the pickup and the target position. This default behavior can be changed by selecting 'fixed' approach, see 6.2.3 - Movement

---

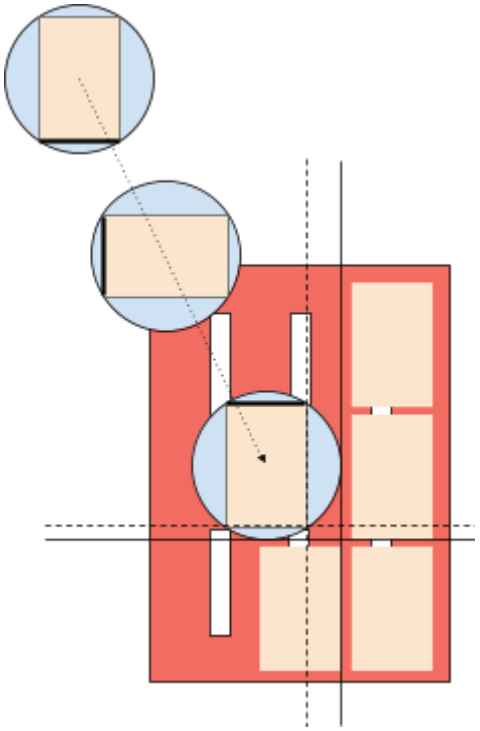


Figure 14: Dynamically calculated approach distance for rotating boxes.

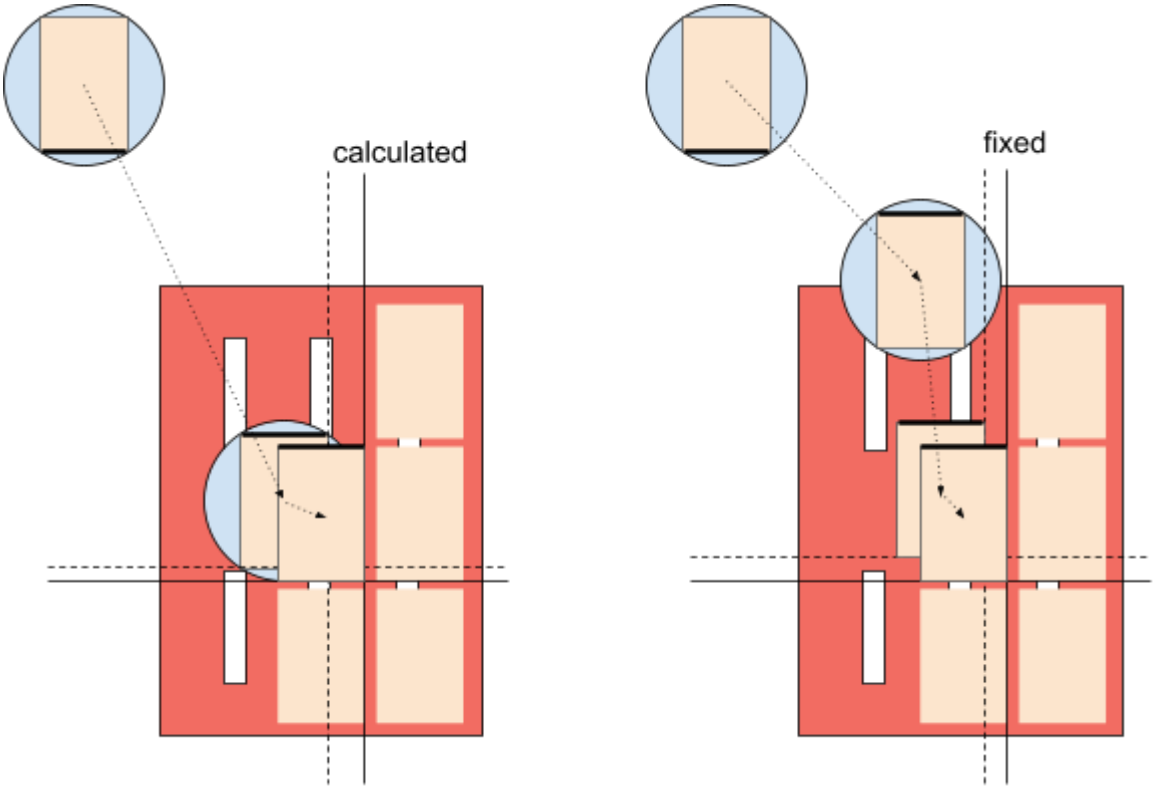
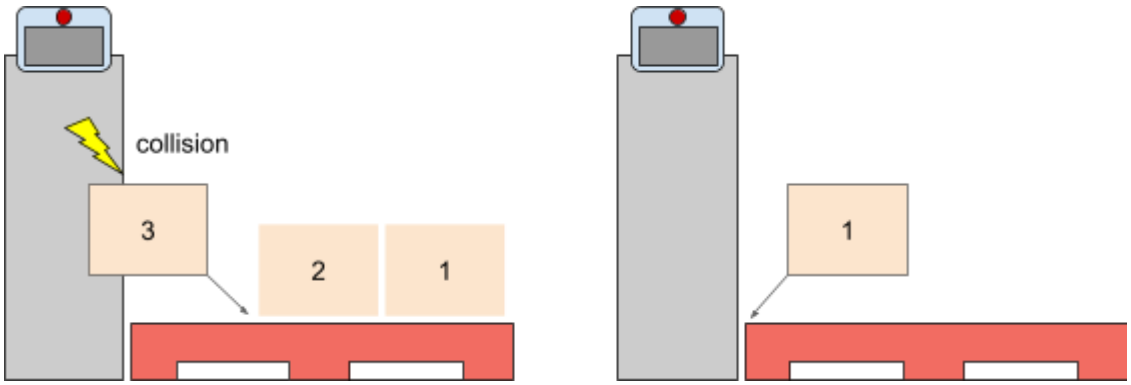


Figure 15: Difference between dynamically calculated and fixed approach distance. Please note the extra waypoint added to complete rotation before approaching.

When palletizing the lower layers, the robot may not have enough room between the mounting base and the box position to keep the approach distance - the box or the robot arm can collide to the base and get damaged.
To work around this problem, it is possible to change the approach direction and consequently the order of boxes in problematic layers.



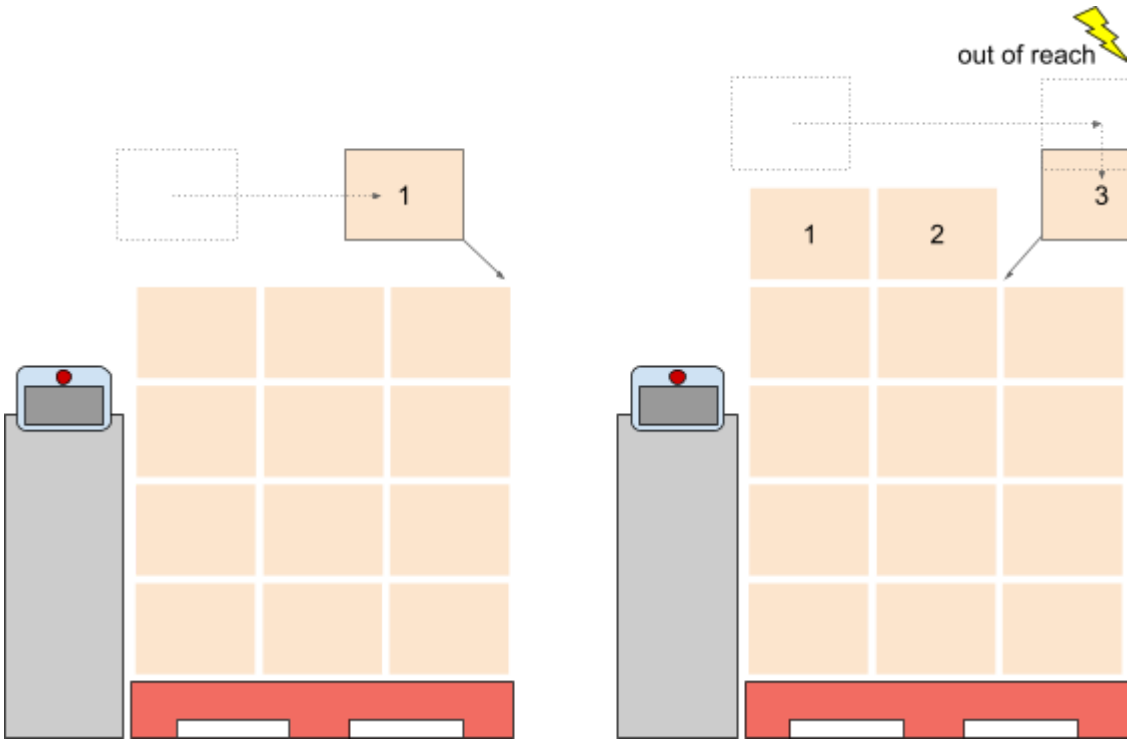


Figure 16: Pros and cons of normal and inverse approach. Notice the impacts on reach, collisions, box order, etc.

4.5.4 - Pallet lip

When pallets have an outer edge lip, the robot can enforce a vertical movement on the first layer to avoid collision with the pallet lip, see **figure 17**.

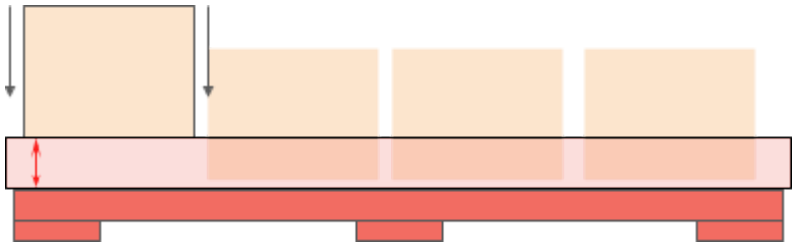


Figure 17: Vertical movement from above the pallet lip

4.5.5 - Return path

After releasing the box at the target position, the robot visits the path waypoints in reverse order, to return from the pallet position and pick the next box(es) from the conveyor.
In a typical setup, the robot waits above the pickup position until the required amount of boxes are available, but in special configurations (e.g. with 2 pickup positions, where the next pickup point is not known in advance) the robot waits at a default waiting position.

It is possible to modify the return path by removing one or more waypoints or implementing a user-defined path.

4.5.6 - User-defined path

It is possible to override the default Pally path planning by implementing custom movement for one or more specific box positions, or even replace the default path planning in the entire project. It is also possible to implement custom code to sort out boxes from the conveyor that should not be palletized, based on a project-specific signal or quality check. Refer to callbacks for further details.

4.5.7 - Path storage offline database - verified patterns

During palletizing a specific pallet pattern for the first time, the program creates an offline database with all waypoints, lifting column positions, and gripper configuration being used for each box position. The database is updated with new data after each successful robot cycle until all box positions are stored.
Subsequent pallets are being performed by reusing the waypoints stored in the database, skipping the calculation routines completely if possible.
The pattern is considered *verified* when the corresponding path storage database contains all the necessary waypoints to palletize full pallet(s) from one or two pickup positions, according to the current setup:

- If both left and right pallets are enabled, both must be tested, and
- If both primary and secondary pick positions are enabled, both must be tested.

The path storage database can be deleted and generated again at any time, as long as the Service and Support plan option is active.

Note: Without a valid Service and Support plan it is only possible to palletize verified patterns, i.e. patterns that have a complete path storage database. Make sure to test all patterns before your Service and Support plan expires.

Note: A pattern is considered verified if it is possible to palletize full pallets, starting from position 0, by using the path storage database only. When palletizing partial pallets of multi-pick patterns however, additional calculations will be performed and the program may even fail with 'out of reach' error at the first or last box positions, which requires single-pick. In such setups it is highly recommended to test multi-pick patterns also in single-pick mode and make sure all box positions are within reach when building partial pallets.

4.6 - Payload control and robot stabilization

Collaborative robots are generally very sensitive to the weight and center of gravity of the attached payload. The robot settings must always correspond to the actual payload in order to work properly. Changing the payload settings too early or too late will result in instant

protective stops due to false collision alarms. Especially in palletizer applications where heavy boxes are picked and released, it is essential to manage payload settings as accurately as possible.
The program has built-in routines for monitoring errors in the robot position, which may occur due to sudden changes in the payload. During picking and releasing boxes with a vacuum gripper, the program can automatically adjust the delays between switching the vacuum valve(s) and updating the payload weight and center of gravity. This can significantly reduce the occurrence of unexpected protective stops at the pick and drop positions without affecting robot safety requirements.

Note: This function may introduce small delays when picking and releasing boxes. Some tuning parameters are provided for adjusting or disabling the function.

4.7 - Shim papers

Shim papers are considered as special layers with no products. These layers have their own height - i.e. the thickness of the shim paper itself. Pally does not include a default implementation for shim paper placement - this has to be implemented for each project by using callbacks.

4.8 - Lifting column

Lifting columns extend the robot reach by adding a virtual 7th axis to the robot. The current position of the lifting column is a parameter in all calculations in the Pally path planning.

Lifting columns can have 2 or more valid positions. 2-position lifting columns are typically built with end-switches (up/down positions) while other models have continuous positioning with position feedback. When continuous positioning is available, Pally will choose intermediate positions that correspond to multiples of layer heights. Otherwise, the lifting column will be elevated when the pallet height reaches the stroke of the column, see **figure 18**.

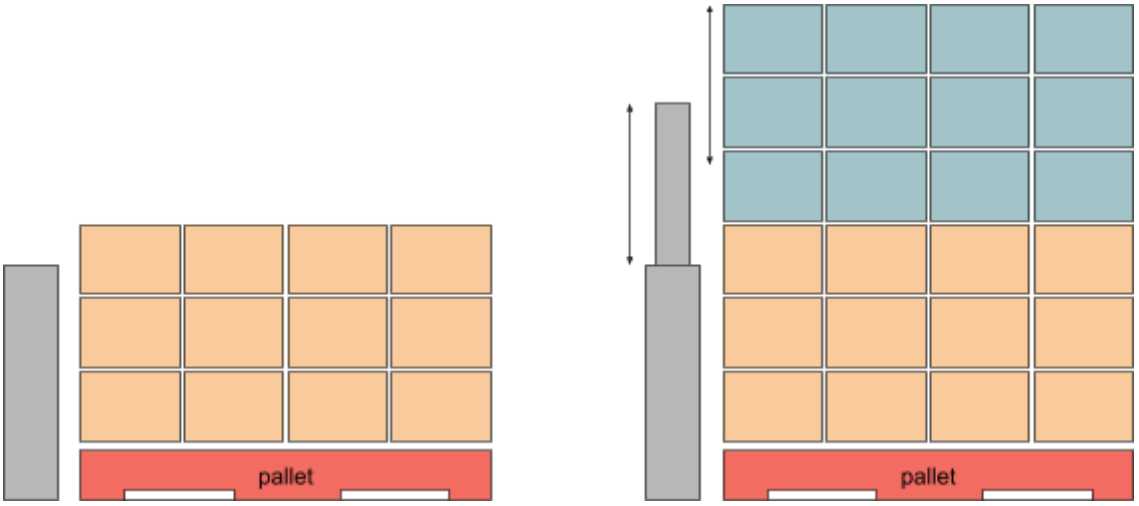


Figure 18: Lifting columns can extend the robot reach in the Z-axis

Each layer is palletized with the best lifting column position by default, but custom configurations are also possible by defining zones manually.

Note: It is expected that the lifting column changes the robot altitude only, i.e. moves along the Z-axis. No other configuration is supported.

4.8.1 - Positioning the lifting column for each layer

In this operational mode, the lifting column position is calculated for each layer or group of layers. The first layer is palletized at position 0, then the lifting column is elevated by the height of one layer until the maximum stroke is reached. However, the maximum number of lifting column operations is limited to 5, which means that more than one layer of products with small product height will be palletized at the same lifting column position before it is being lifted up.
When using this mode, the pickup position must be within reach from all lifting column positions, otherwise the dynamic positioning option is required.

4.8.2 - Positioning the lifting column for each box - dynamic positioning

In this operational mode, the lifting column stroke is calculated for each pick and place movement individually. This is often necessary in projects where the total stroke of the lifting column is large and/or the pickup position can go out of reach at some point (e.g. very high pallets). When using this mode, the lifting column position can be different at pick and place. First, the algorithm calculates the lifting column position range from where the pick position is reachable (lowest and highest column position) then the same calculations are performed for the pallet target position. If the two ranges have a common intersection, a lifting column position is chosen from the common range i.e. the same position will be used for picking and placing the box. If no intersection is found, two lifting column positions will be chosen for the

pick and place positions respectively. Refer to the Lifting column tuning parameters for prioritizing which side of the available range the program should prefer (e.g. keep the lifting column as low as possible, as high as possible, keep it in the center of the possible range)

Figure 19: selecting the best lifting column position

4.8.3 - Custom positioning - zones

It is possible to override the default behavior and take full control of the lifting column positioning by creating a special configuration file. In this case the pallet will be split into smaller regions and each region palletized at different lifting column positions. For full flexibility, each pattern can have its own configuration, and optionally a global configuration can be applied for all the remaining patterns that don't have their own settings.
See zones for further information.
Please note that this mode cannot be used in combination with dynamic positioning.

4.9 - Zones

Normally the robot completes a layer before starting the next layer. However, the pallet can be split into different regions - called zones - along the X, Y, and Z-axis. As a general rule, a zone cannot be started until the previous zone is finished.

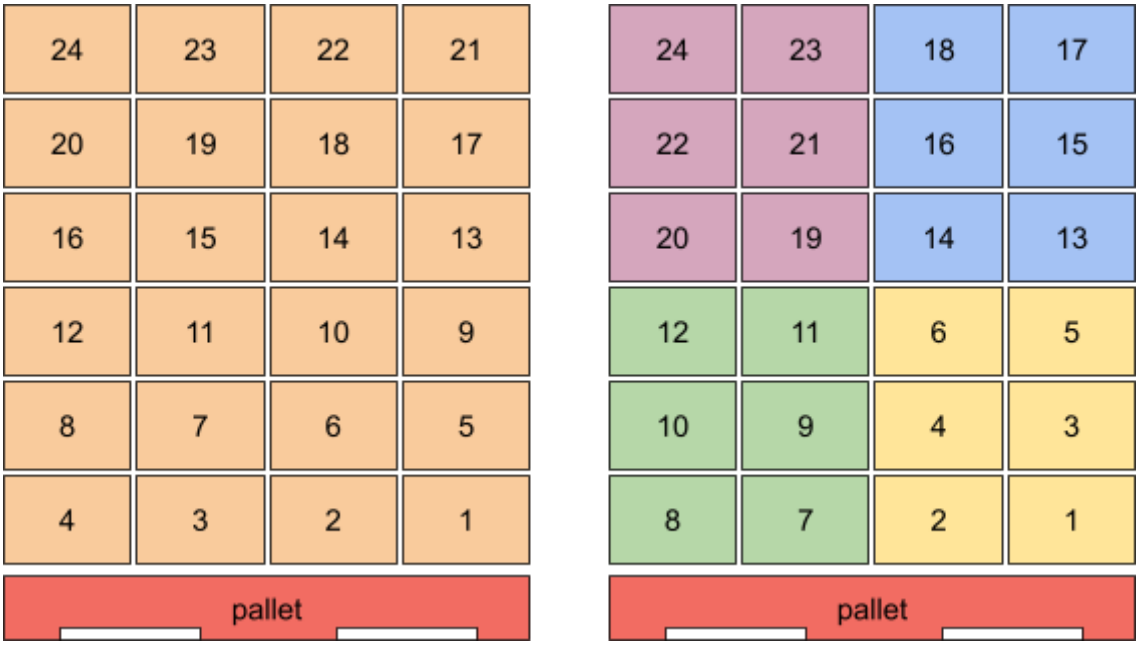


Figure 20: illustration of how zones divide pallets into smaller regions.
Each color indicates a different zone.

Zones is one of the most powerful features and can be used to solve problems that could not have been done otherwise, e.g.:

- Avoid collision between upper arm and boxes nearest to the robot on the top layers,
- Take full control of the lifting column positioning,
- Control external machines, for example, start a stretch wrapper to wrap the pallet after N layers are completed.

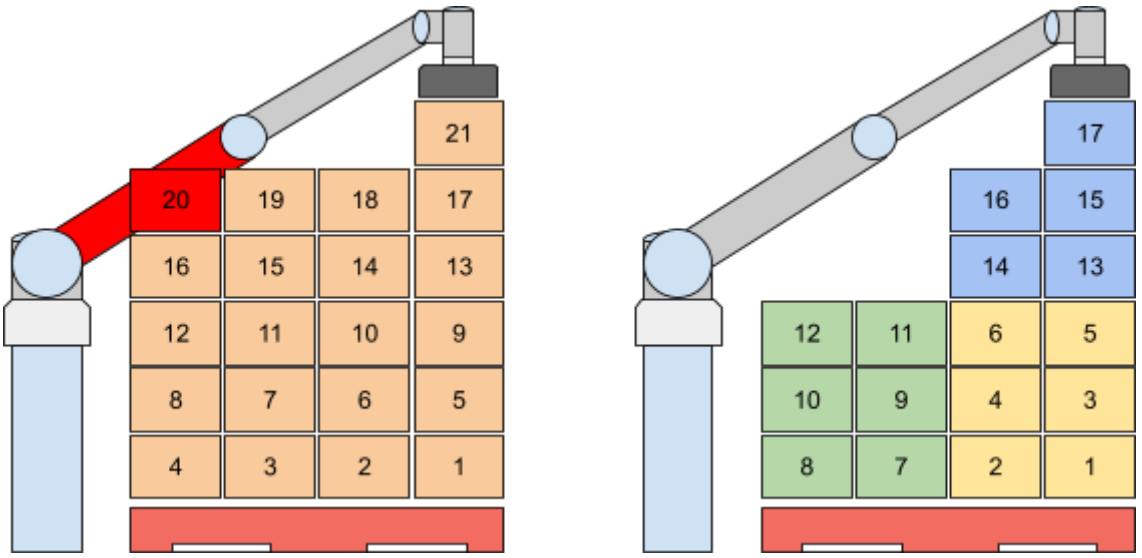


Figure 21: using zones to avoid collision by changing the palletizing order.

Zones require a special configuration file. This file will describe the conditions for splitting the pallet into smaller regions. Creating the configuration file is easy and can be done in most cases by copying the example files provided.
See the definition of Zones for more details.
The zone configurations are transferred to the robot along with the regular pallet patterns, see Uploading new patterns.

4.10 - User extensions - Callbacks

It is possible to extend the functionality with user-defined commands, which are automatically executed when specific events occur.
Currently the following events are available:

- The initial MoveJ node: Executed once on startup, after all configuration variables have been initialized. This allows runtime configuration changes and special tuning.
- **beforePallet**: Before starting a new empty pallet.
- **beforeZone**: Before entering a new zone.
- **onNextTask**: Before starting all calculations for the next box(es)
- **beforeGrab**: Before lifting up a box from the pickup position.
- **afterGrab**: After lifting up a box from the pickup position.
- **beforeRelease**: Before releasing a box on the target position.
- **afterRelease**: After releasing a box on the target position.
- **afterZone**: After leaving a completed zone.
- **onSheet**: When a shim paper needs to be inserted.
- **afterPallet**: After the pallet is complete.

The user-defined commands can utilize the values of the following global variables:

- **ProductCount**: the number of boxes palletized on the pallet so far. The value is incremented between the beforeRelease and afterRelease callbacks if TaskCompleted is True (see below)
- **ProductName**: name of the product that is being palletized
- **PalletNr**: 1=right pallet, 2=left pallet
- **PalletCenter**: a pose that defines the position and orientation of the pallet top center
- **ZoneNr**: indicates which part of the pallet is being palletized (for lifting columns), 0-based
- **LayerNr**: indicates which layer is being palletized, 0-based
- **ProductHeight**: height of the box that is being palletized (in meters)
- **LayerHeight**: height of the current layer (in meters) - equivalent to ProductHeight in callbacks beforeGrab, afterGrab, beforeRelease, afterRelease. Contains the height of the current shim paper in the callback onSheet. Otherwise zero.
- **LayerAlt**: the altitude, measured from the top of the empty pallet (for shim papers)
- **WorldPosition**: a pose that defines an extra transformation for elevated lifting column

- **MoveTarget**: target position of the robot (pose, p[x, y, z, rx, ry, rz]). In the afterGrab callback, this variable contains the target pose of the box on the pallet where the robot should move. In the afterRelease callback it contains the pose of the default waiting position, transformed by the lifting column position. Use this variable in combination with MountPosition when using a lifting column.
- **MountPosition**: expected mounting position (i.e. lifting column position) at the moment when the robot is at MoveTarget. When using a "custom path" with a lifting column in Dynamic Positioning mode, it is required to control the lifting column position from the callback code along with the robot position. The value of MountPosition is an array with 3 elements [x, y, z] where the first two values are always zero and reserved for further extensions; the third value is the expected lifting column height at the moment when the robot is at the MoveTarget position.
- **MovePerformed**: this is a user writable variable to indicate that the robot has been moved to the MoveTarget position and the program should skip the movement. With this variable it is possible to implement custom movement from the pickup to the target position, or back from the target position to the waiting position. Set this variable to True in either afterGrab or afterRelease, depending on the direction of the movement that has been performed by user code.
- **TaskCompleted**: this is a read-write boolean variable to indicate that the box(es) are successfully palletized and the box counter can be incremented. The default value is normally True, but can be False when a vacuum sensor is installed and vacuum is not detected at the target position before releasing the boxes. Set this variable to False to repeat the same box position, e.g. when a project-specific quality check is implemented and some boxes are moved from the pickup position to a waste bin instead of being palletized.

Some typical examples of uses for user-defined commands are to insert a shim paper, start and stop the conveyor, sort out inappropriate boxes to a waste bin, perform custom motion in very special layouts, control light signals and signal external machines, etc.

4.11 - Manual and automated operation modes

The palletizer program can be controlled manually by the operator, or automated via the corresponding variables and interfaces.
When using the program in manual mode, the operator can select the pattern to be palletized, optionally choose the number of boxes and define the starting position on the pallet.
In automated mode, it is possible to start the palletizer program with a predefined pattern, or let the operator select from a list of patterns. There are currently 3 different strategies to select patterns:

- Automatically select the (alphabetically) first pattern from the pattern storage folder
- Automatically select the pattern with a specific name, as defined in a special variable
- Manually select the pattern by the operator

For further information, see the configuration variables in the Automation section.

4.12 - Order management (beta)

It is possible to palletize batches of full and partial pallets by creating an order list in advance. Orders can be created manually by the operator, or automated through API functions. In automated mode multiple orders with different patterns can be added without stopping and starting the main program. The program will wait for new orders and perform the orders until stopped. In manual mode however, it is only possible to create one order with one or more full and partial pallets, after selecting the pattern by the operator. For further information see the order management section.

4.13 - Show mode

The Show Mode demo has been created to show the different functionalities of Pally. It will depalletize from one pallet, drop the boxes off on the conveyor/drop-off position and then palletize them on the other pallet.

For the Show Mode demo program to work, the following is required:

- two pallets
- a conveyor or gravity slider (recommended)
- one of the two following sensor options:
    - Two sensors, one for confirming drop off and one for pick up (recommended)
    - One sensor alternating between signals; “High” for box presence on the conveyor (palletizing) and “Low” for conveyor empty (depalletizing)

The robot will start with depalletizing from the full pallet, and put the box on the drop position on the conveyor, which is further away from the pick position. Then the box will travel to the end of the conveyor and get aligned perfectly. The pick signal is triggered when the box arrives. The robot will pick the box and palletize it to the other pallet. When all boxes are moved, the program will start depalletizing from the other pallet back, and this is repeated forever.

5 - Physical Installation - Best Practice

5.1 - Robot installation

5.1.1 - Check joint positions first

The UR10 joints have a limited +-360 degrees position range, so it is essential to check whether the joint positions are not too close to their limits, before attaching the vacuum hose and other objects (**figure 22**). Having a bad initial position may cause the robot to stop with *Joint position close to the limit* during palletizing.

Note: The robot can be operated as "left-handed" or "right-handed". Left handed or right handed implies on which side the shoulder joint is located, seen from the opposite side of the conveyor. For a right handed robot, the shoulder joint is located to the right side of the base joint. For a left handed robot, the shoulder joint is located to the left of the base joint. The examples in this document are made with a left-handed configuration.

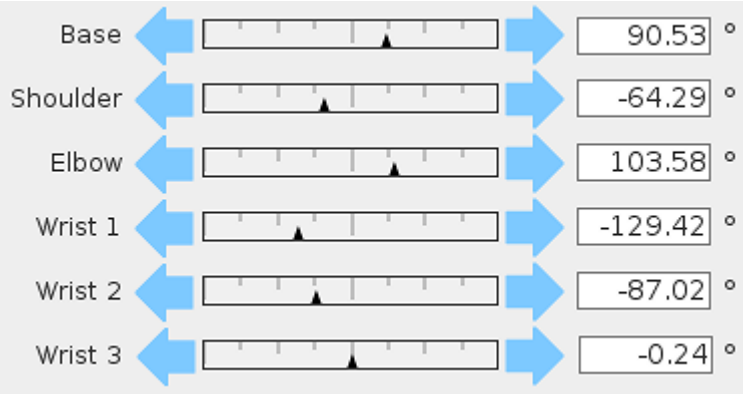


Figure 22: Check the joint positions before mounting vacuum hoses and cables.

5.1.2 - Robot position on the base column

Before mounting the robot on the base column, make sure that the power connector on the robot arm points behind the palletizer cell, see **figure 23**. This is not mandatory, but recommended for best reachability both on the left and right pallet positions.

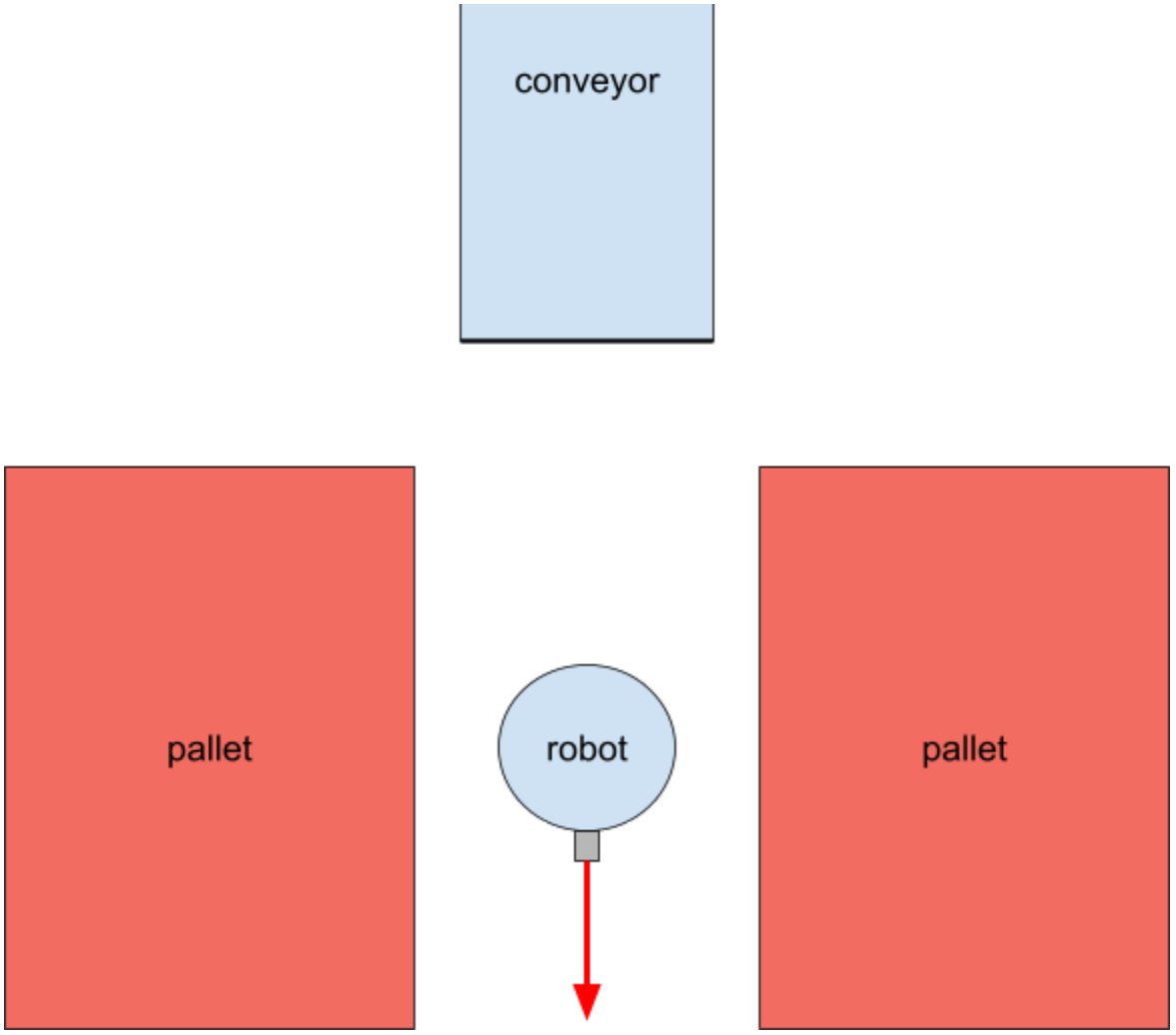


Figure 23: Check the position of the power connector.

5.1.3 - Attach the gripper properly

Attach the gripper only when wrist 3 is close to its zero position. This way you can maximize the rotation range in both directions.
Depending on the physical dimensions of your gripper, you might have to configure the robot tool center point under Installation / General / TCP as described below.

Note: The conveyor direction is calculated directly from the Y-axis of the tool coordinate system at the pickup position, as shown in **figure 24**.

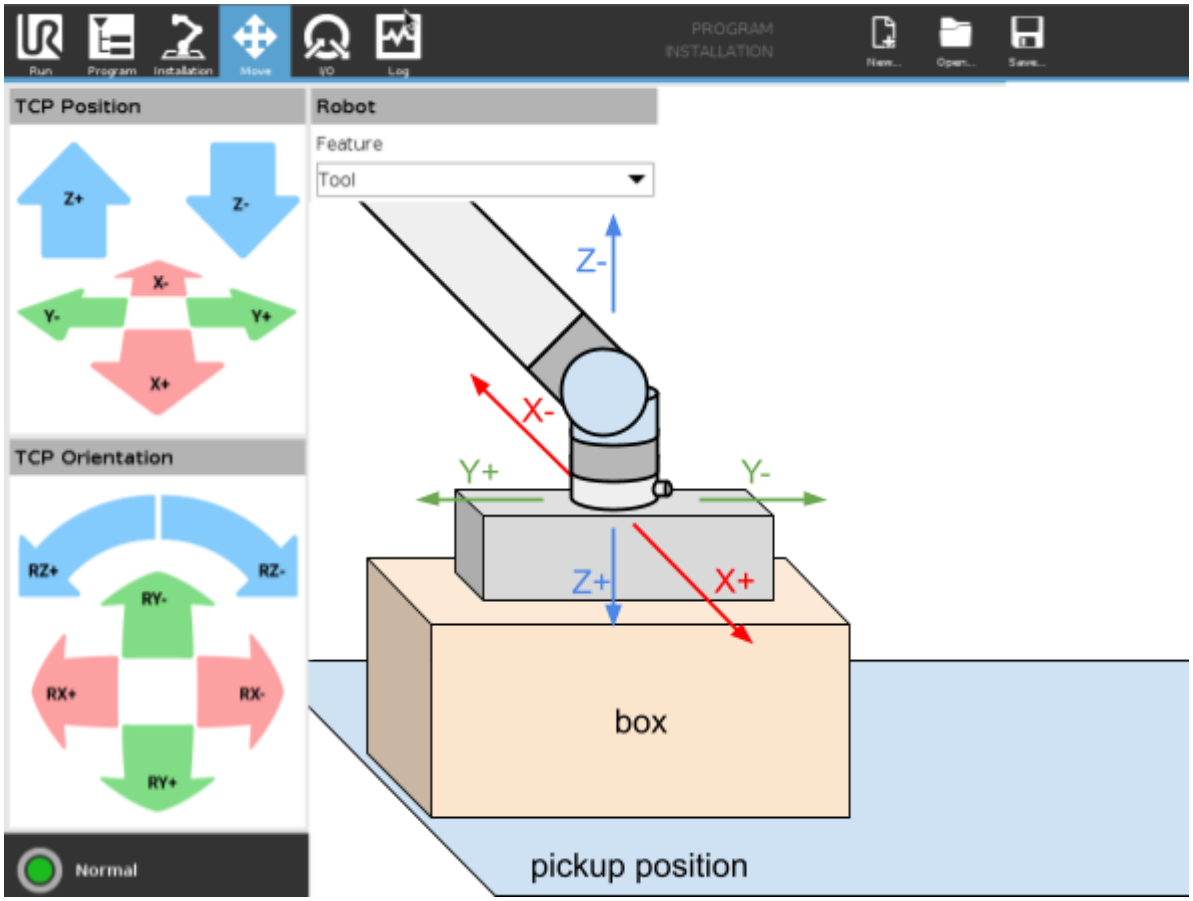


Figure 24: Make sure TCP is configured properly.

Pally supports a wide range of grippers, and some of them might have either position, rotation, or both offsets that must be taken into account when TCP is being configured. To verify the TCP settings are correct, perform the following steps:

- Move the robot to the pickup position.
- Now go to the Move tab, and select "Tool" in the drop-down called "Feature".
- Now press the Y- arrow button and the gripper should move parallel to the conveyor and in the direction where boxes arrive from, i.e towards box 2.
- Press the Y+ button, and the gripper should move back to the end of the conveyor.
- When the gripper is above the box on the pickup› position, press the RZ+ and RZ- buttons: the gripper should rotate around its own center point and not around the robot tool flange. Rotating by 180 degrees, the gripper should remain perfectly above the center of the box at the pickup position.

Note: Setting the TCP properly is essential for multi-pick and other path planning calculations.

5.1.4 - Lifting column

Ignore the following steps when using the lifting column with dynamic positioning.

- Make sure the pickup point is reachable when the lifting column is in its retracted (low) and stretched (high) position.
- In projects with multiple products: repeat the previous step with the smallest and the largest expected products.
- In projects with multi-picking: repeat the previous steps with 2 (3, 4, etc.) products.

5.2 - Layout

The palletizer cell consists of the robot, the pallets, and the pickup position. The conveyor and the pallet positions must be calibrated by moving the robot (with the gripper attached) into various calibration points as shown in **figure 25**.

Note: Always use the "width" and "length" values as shown below. Make sure that the width and length values are used consistently during the entire calibration.

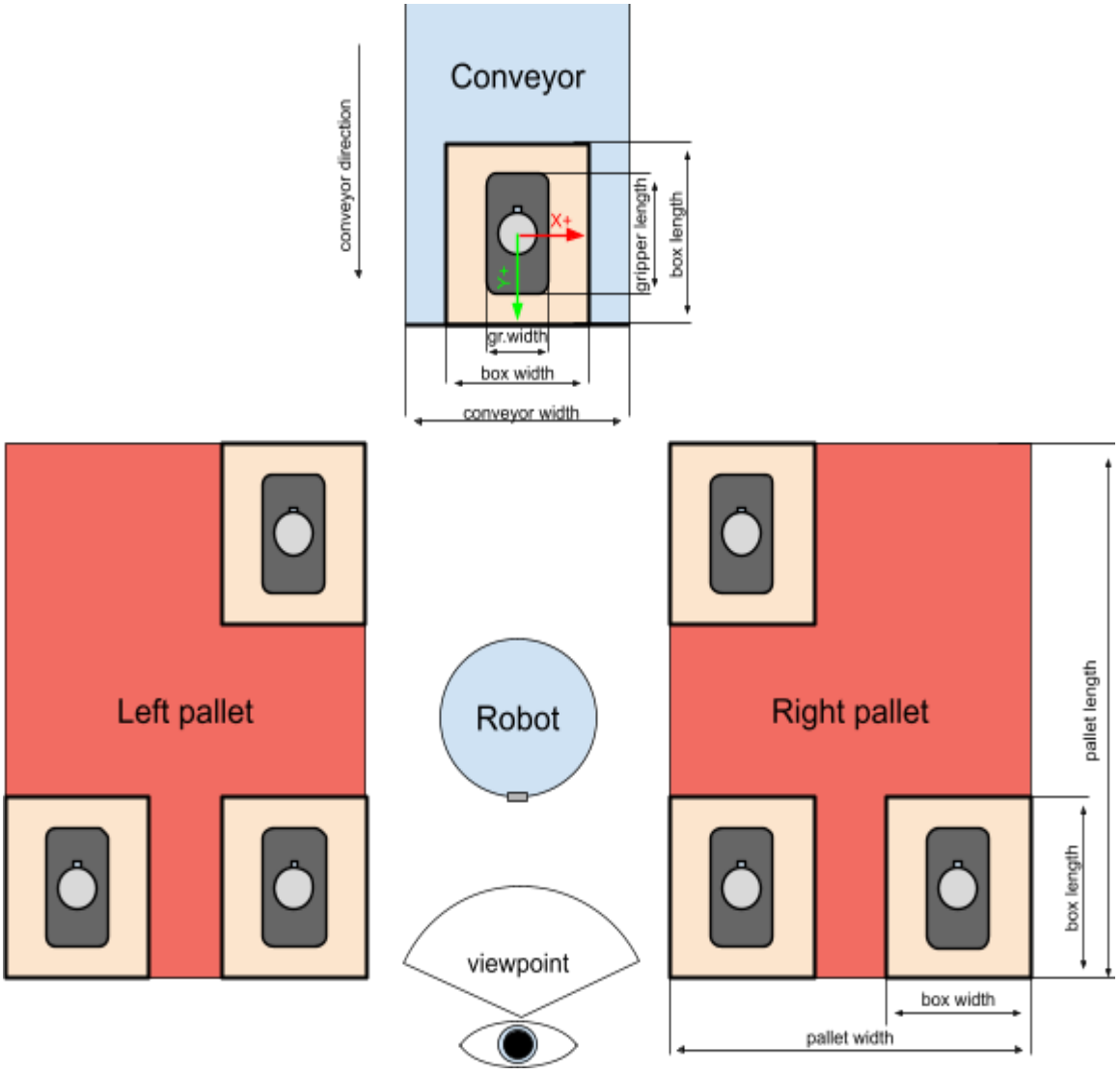


Figure 25: Calibration points for the pickup position and pallets.

5.2.1 - Supported pickup positions

Product arrival angle

Pally supports several different pickup positions. The boxes may arrive at the pickup position from the side, from the front or from an angle. I.e., the boxes can arrive both sideways or straight towards the robot as illustrated by the arrows in (**figure 26**). The end point for the pickup must be a fixed position.

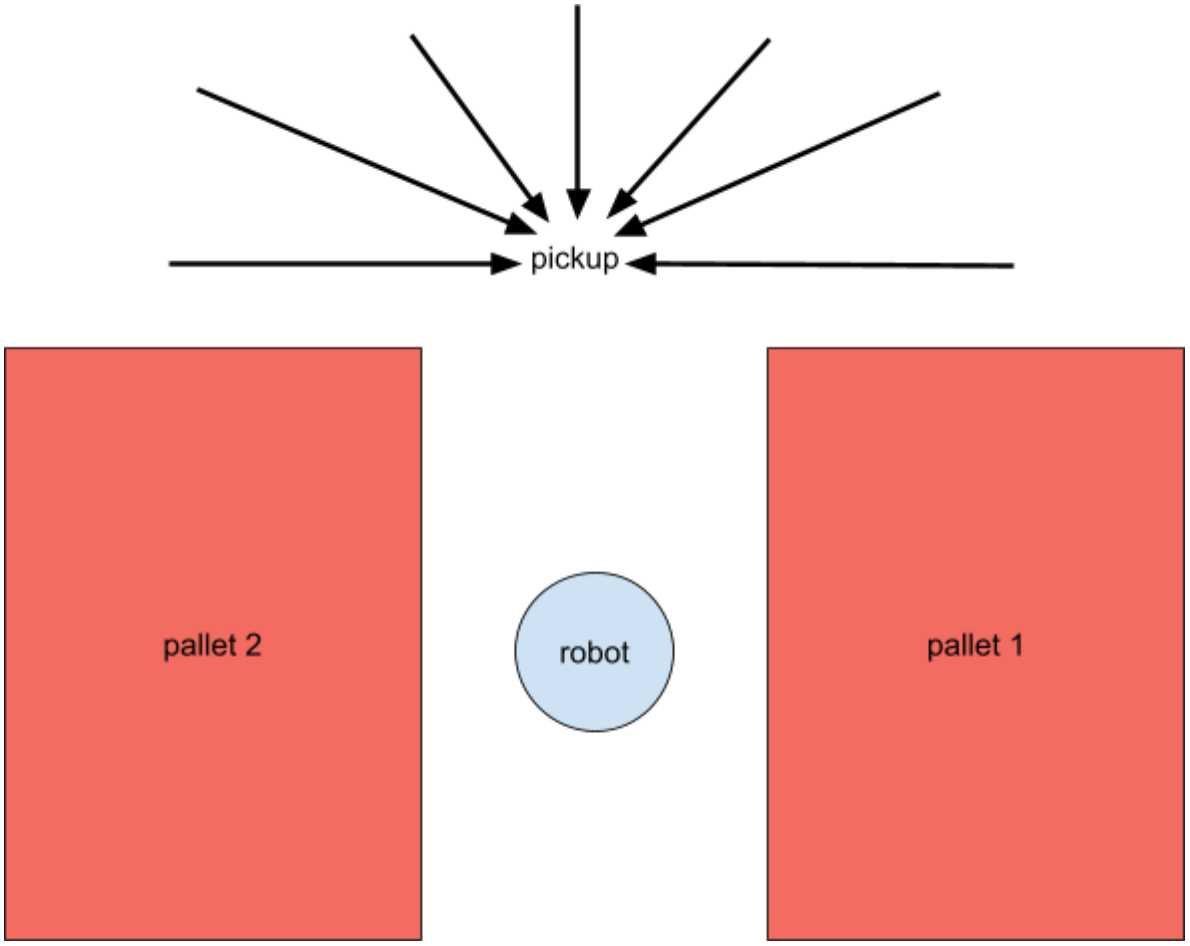


Figure 26: A product may arrive at the pickup position from a wide variety of angles, but should always be stationary on the same location when being picked up by the robot.

Multi-picking

Pally supports lifting and placing two or more products at the same time, as long as the total payload and the placement on the pallet allows it. When using multi-picking, the boxes must be aligned on the conveyor. Either guides or pushers are recommended for this purpose. Low guides are preferred, since these will give more room for the robot to optimize its paths for increased capacity and increased lifespan. This is illustrated by *"box free"* in **figure 27**.

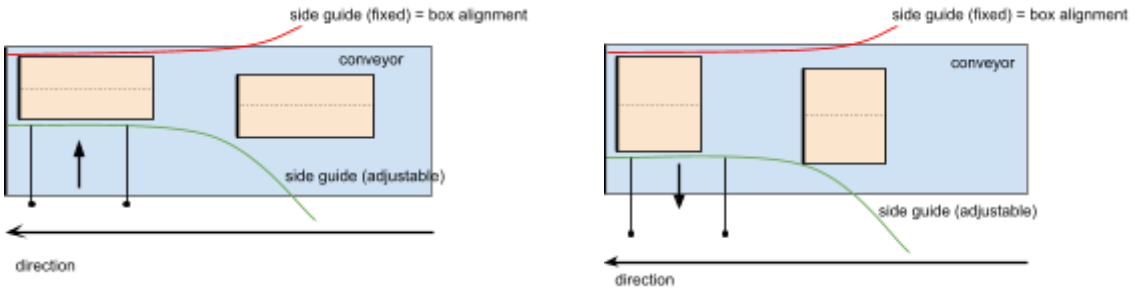


Figure 27 : Use side guides or pushers to align products to the same reference point.

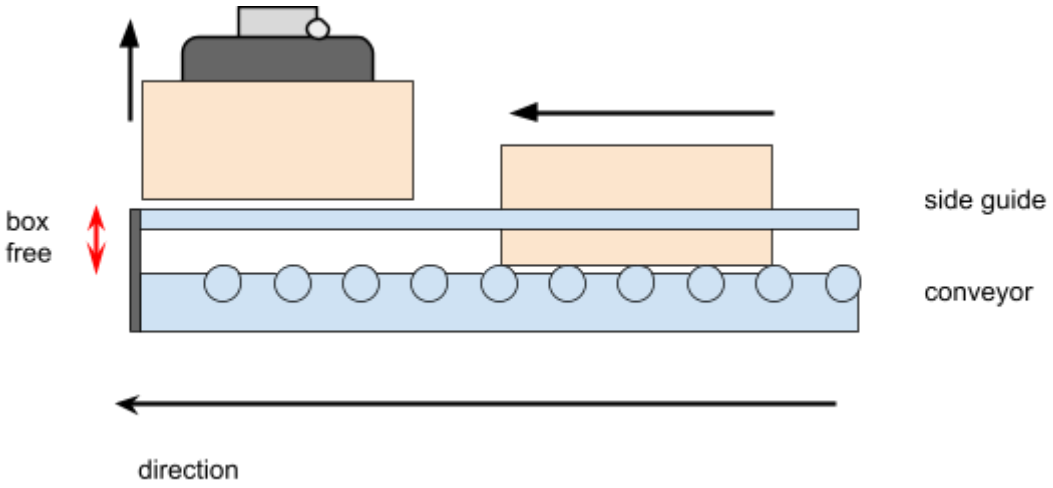


Figure 28: Shorter “box free” travel will improve speed and expand robot lifetime.

Two pickup positions

When picking from two lines simultaneously, it is recommended that the two lines come into the robot parallel to the y-axis. The robot has less free space to move in, and lines coming in from the side may cause crashes when the pallets are nearing fully stacked (**figure 29**).

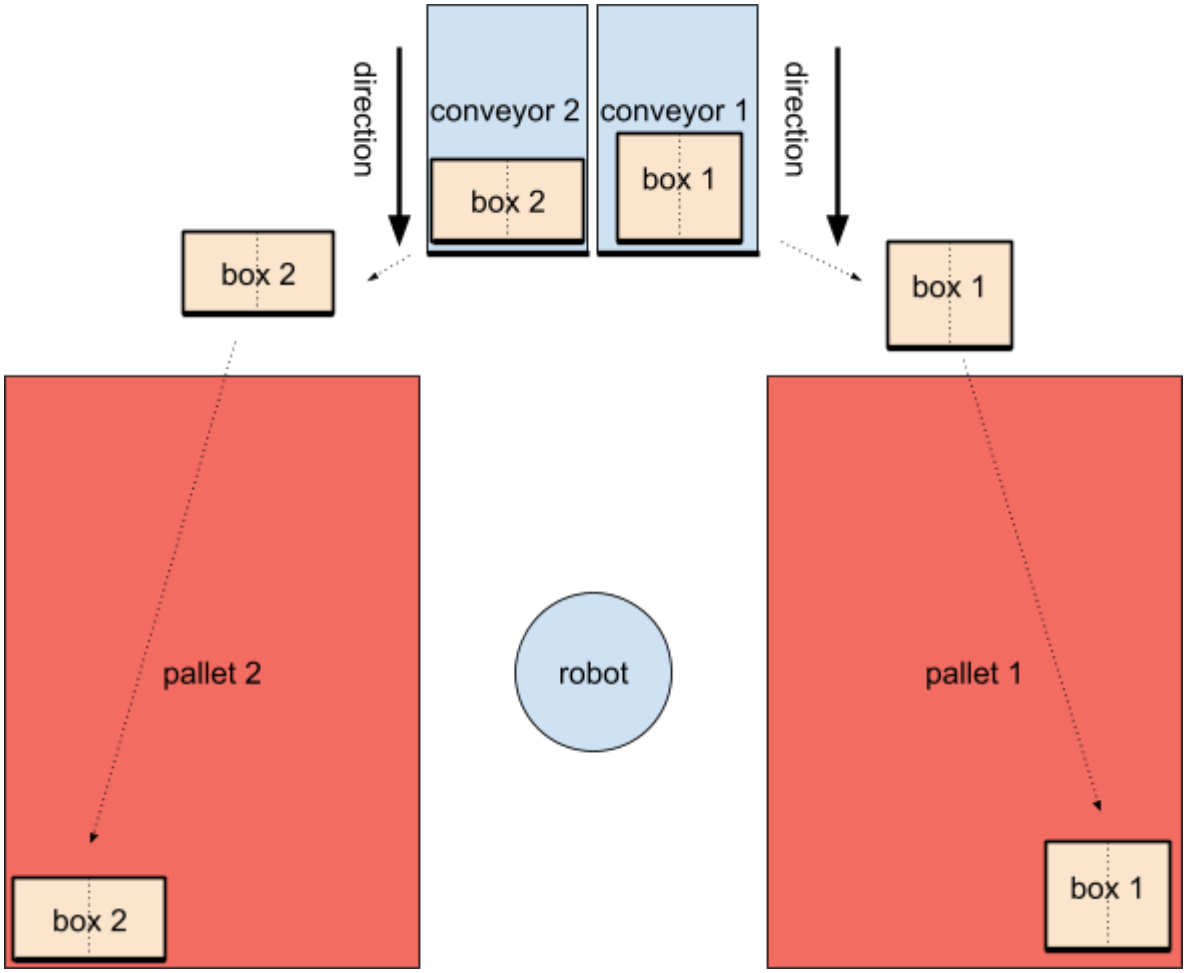


Figure 29: This layout is recommended for dual conveyor solutions. The layout will minimize unnecessary vertical movements for the robot.

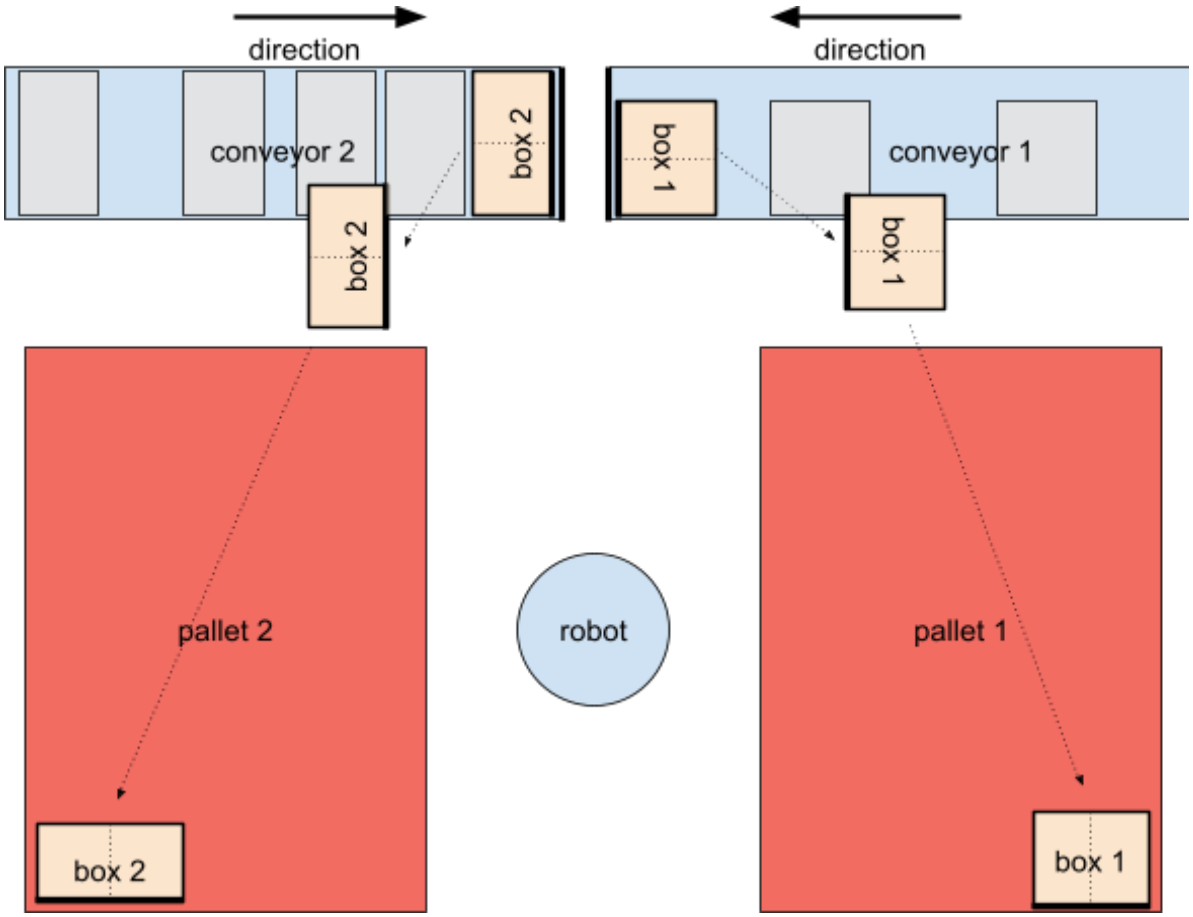


Figure 30: This layout is not optimal since the robot must lift the product *above* the next products on the conveyor to avoid collision.

5.2.2 - Suggested sensor placements

Product presence sensors
Place the sensors as low as possible on the conveyor. Install one sensor at the very end of the conveyor to detect one box ready for pickup. Install the next sensor where the second box will be located (**figure 31**).

Note: If palletizing many different products from the same line, an adjustable mounting of sensor 2 is recommended, so that it easily can be moved according to product size.

Priority sensors
It is recommended to install a sensor close to where the line is full of products waiting to be palletized, to inform the robot that the buffer is getting close to full. This enables the robot to prioritize between lines.

Note: The line should be able to buffer at least 4 products.

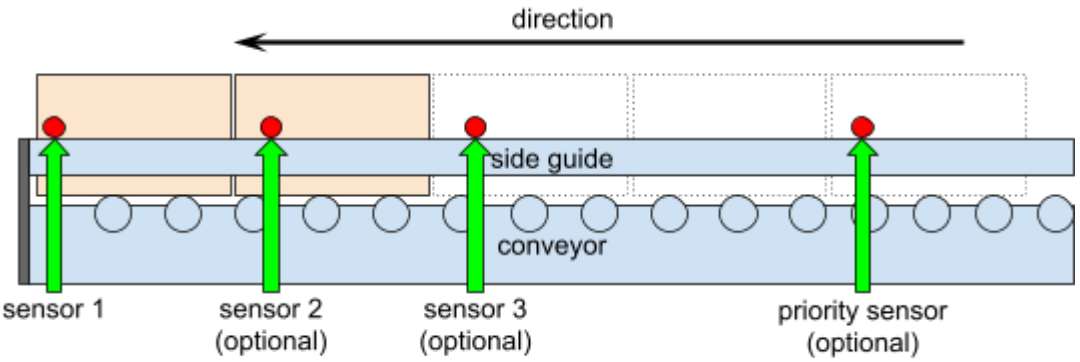


Figure 31: Sensor placement.

Note: Maximum 8 product sensors per pickup position are supported.

5.3 - Dimensions

The recommended dimensions are specified in this chapter. The robot may palletize without issues using other dimensions as well, although significant deviations from the optimal dimensions may introduce unexpected failures.

5.3.1 - Pallet

Pally is compatible with *any* pallets, as long as the selected robot has enough reach across the pallet. Please configure the pallet dimensions in the Pally Pallet Builder (https://palletizer.rocketfarm.no/home).

Pally can palletize on two separate pallets.

5.3.2 - Pickup from Conveyor

For maximum reach and flexibility it is recommended to place the robot so that the pickup position will be approximately at the following coordinates in respect to the bottom of the robot base. See illustrations in **figure 32** and **figure 33** for a standard UR10 setup with EUR pallets and no lifting column.

Please make sure that these values are appropriate for the set up in use. These measurements may vary depending on the lifting column.

Offset sideways	0 mm
Distance from robot to conveyor along the axis of the power cord.	800 mm
Height from ground to the pickup position	700 mm

Note: It is possible to deviate from these values, but it is not recommended to do so by more than ±100 mm.

When referring to the length and width of a product, the width is the face towards the end of the conveyor, and length is the face parallel to the conveyor direction (**figure 34**).

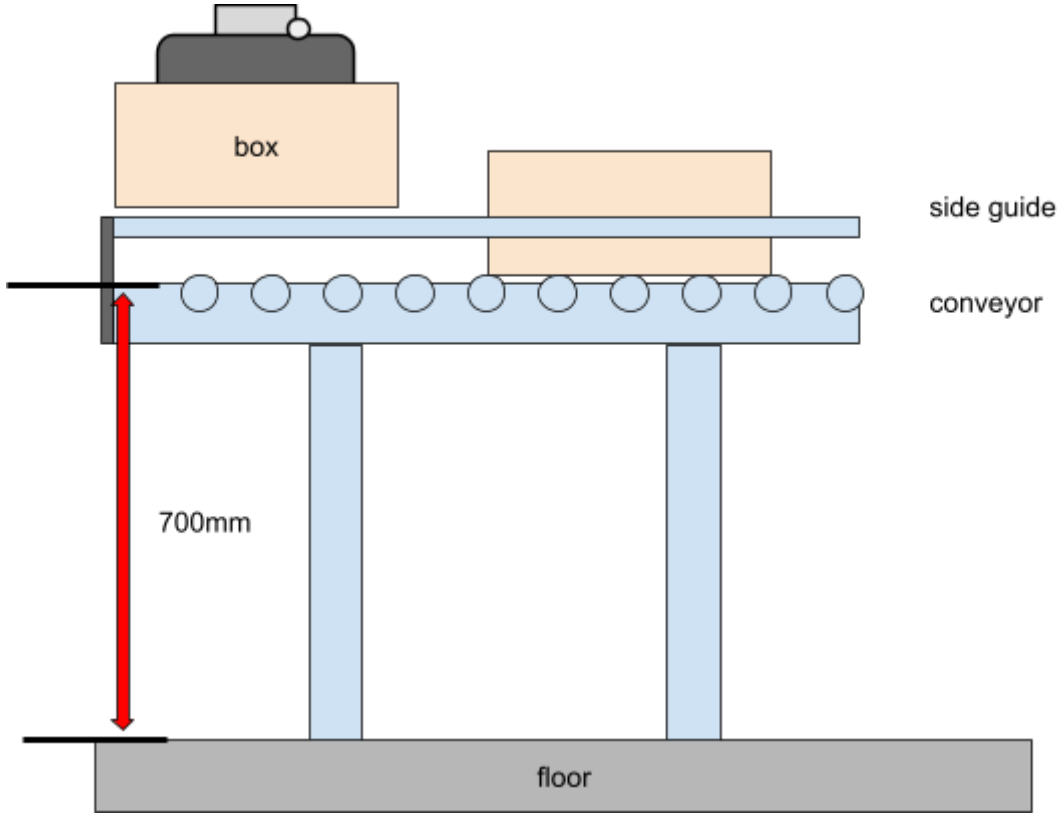


Figure 32: Recommended height of the pickup location.

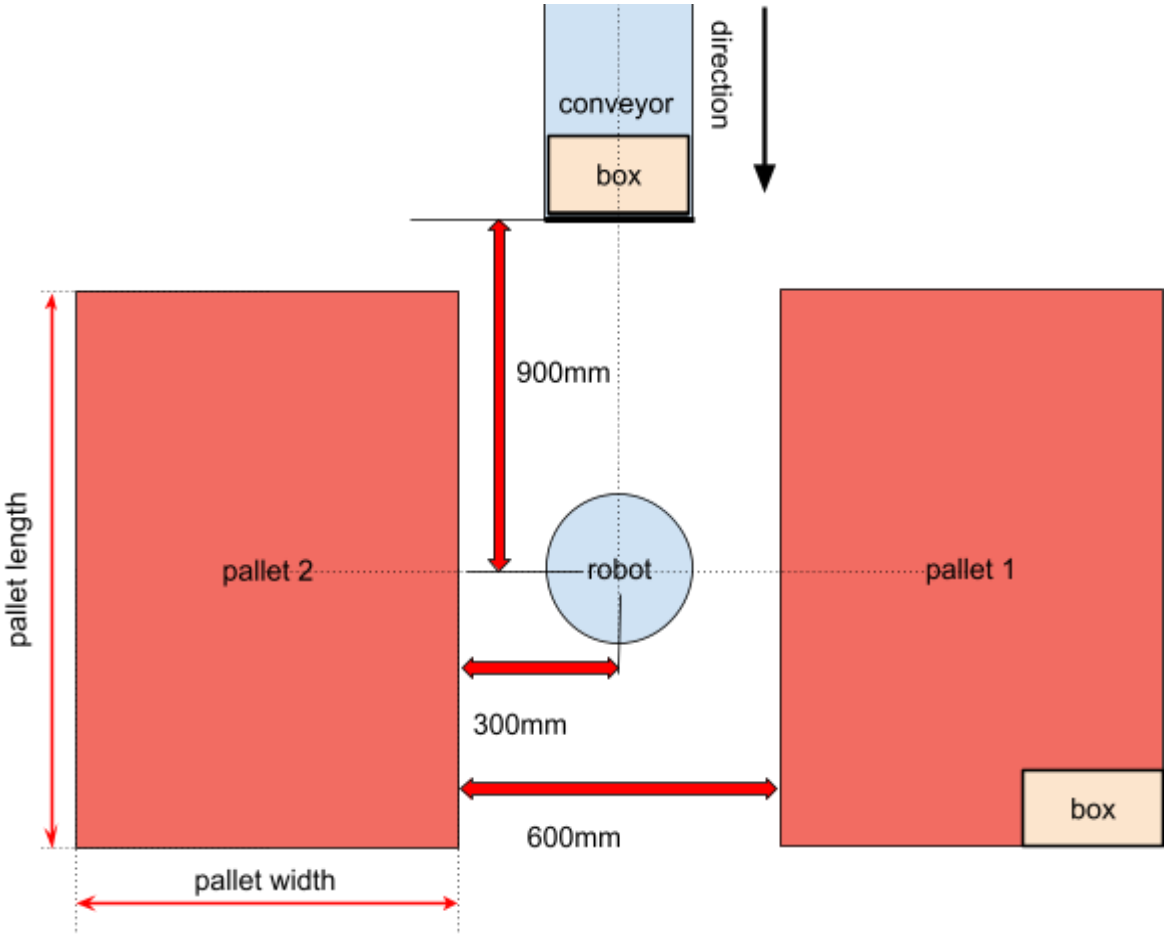


Figure 33 : Recommended placements of components relative to the robot base (UR10)

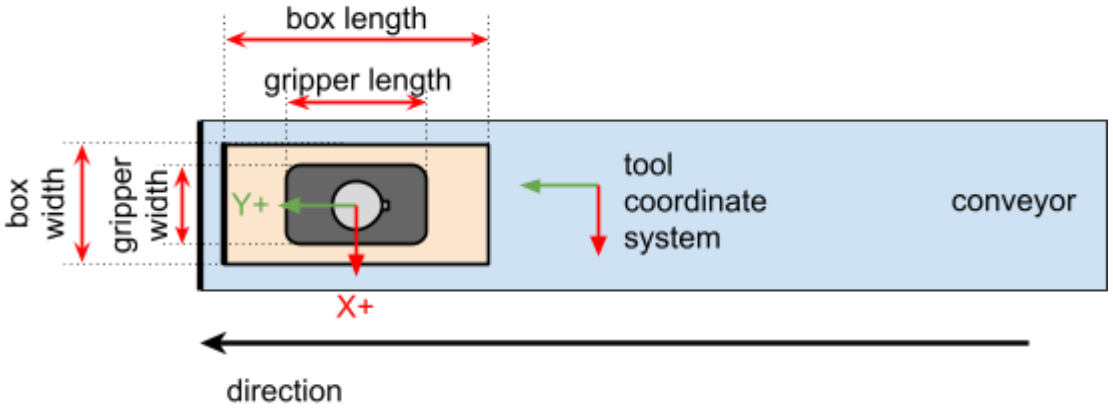


Figure 34: Box and gripper orientation relative to conveyor direction.

5.3.3 - Important parameters to be measured

The path planning algorithm requires the following values to be measured and entered:

- The total width of the conveyor, including any mounted objects. E.g. a motor.
- The position of the fixed guide. Left or right side of the conveyor, seen from the end.
- The fixed guide width, including any objects mounted on the conveyor. E.g. sensors.
- The minimum elevation needed to freely move the box above the conveyor. Measure this from the highest point, including any object which may be a hurdle.

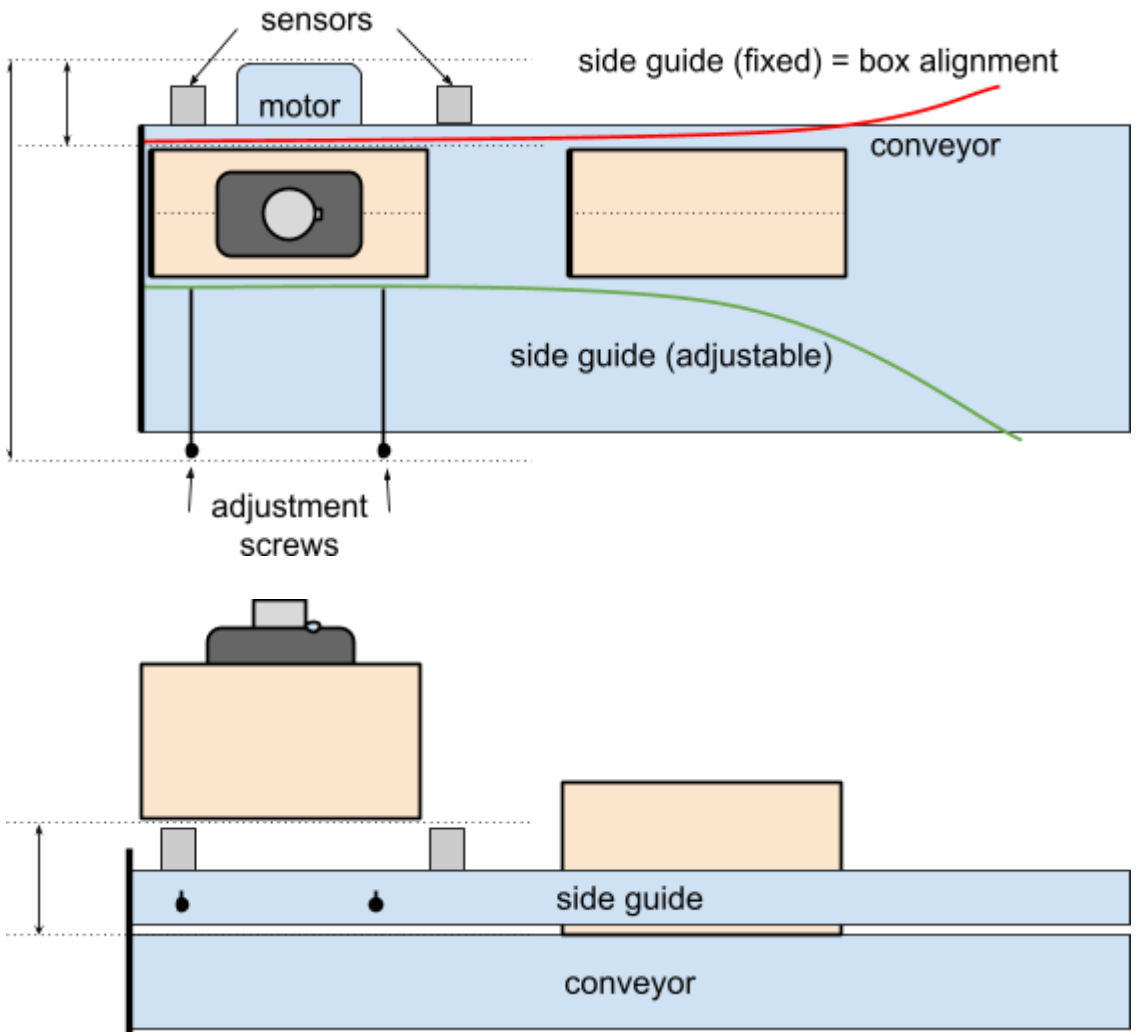


Figure 35: Important dimensions to be measured.

5.4 - Gripper

Pally has built-in support for *UniGripper Co/Light Regular*, *Schmalz FXCB*, and *Piab CPT* grippers, but custom grippers can be also specified. The weight of the gripper is part of the

payload the robot must carry, i.e. a heavy gripper reduces the maximum weight of products to be eligible for palletizing. Keep this in mind when selecting a gripper.

Note: When installing a gripper, the tool center point (TCP) must be configured as illustrated in **figure 34**. Make sure the Y axis of the tool coordinate system is parallel to the conveyor transport direction.

5.5 - I/O Connections

All I/O ports are configurable in Pally, and may be used as needed. Typical configurations, such as a push button to confirm that a new, empty pallet is ready to use, and a light to indicate that the robot knows there is an empty pallet present, are supported.

Note: The program can be expanded to use more I/O in the callback features described in 6.3 - Callbacks.

---

6 - Installing and Configuring the URCap

Installing the Pally URCap can be divided into three parts; the installation settings, the program settings and the callbacks for user customization.

Note: Install the URCap according to the software manual for your Universal Robot before proceeding.

6.1 - Installation settings

This chapter describes the various settings in the installation routine.

Note: This is a one time procedure, and it is important to follow the steps precisely.

6.1.1 - Overview

The Overview tab shows the completion status of the required installation settings. Verify that all items listed in this section are properly configured.

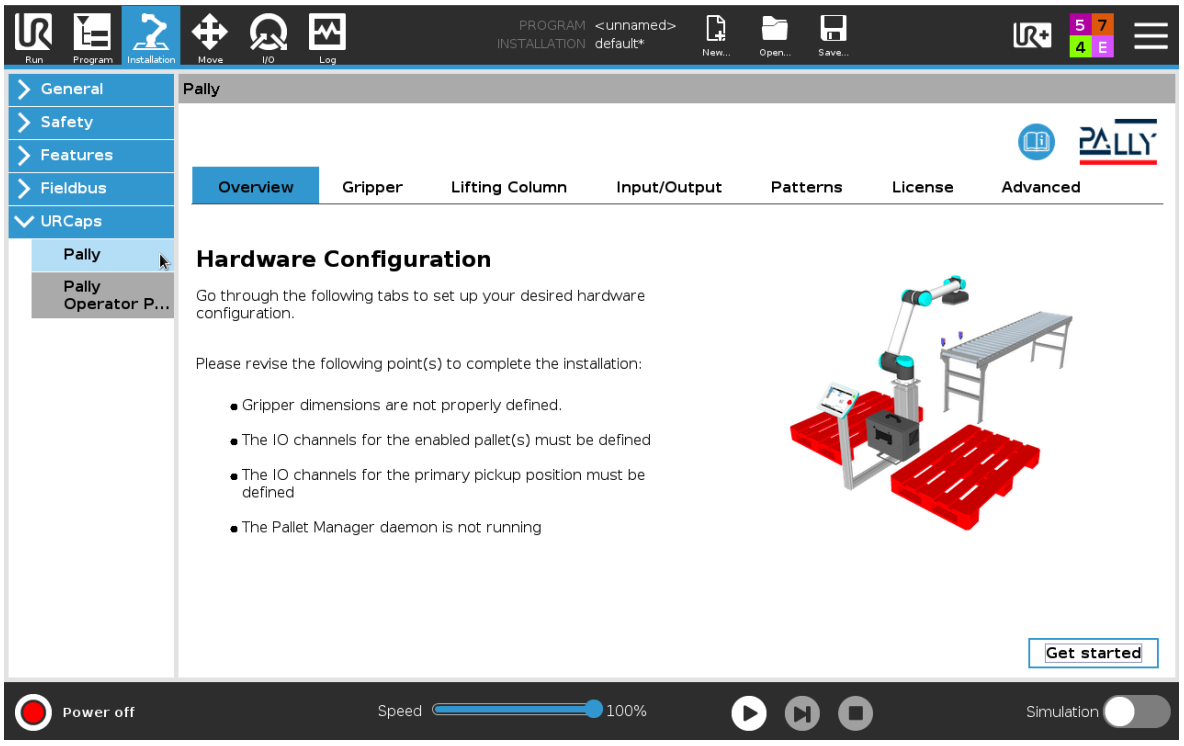


Figure 36 Overview of the installation status

6.1.2 - License

Using the program requires a valid license. Licenses can be installed and removed on the License page.

6.1.2.1 - Installing a license

A valid license must be obtained before the Pally URCap can be used. This can be done as follows:

- In the Pally Installation node, navigate to the tab labeled ‘License’
- Insert a USB stick into the USB connector on the teach-pendant, click the ‘*Copy request to USB’*-button.
- Send the license request to license@rocketfarm.no
- While your request is being processed, you can continue using the URCap for 72 hours. This option can be activated when the license request has been transferred to the USB.
- Download the license file to the root folder of a USB stick.
- Insert the USB stick in the USB connector on the teach pendant and click the ‘Load *license from USB’*-button.

Note: The 72-hours instant license can be activated only once.

The license request file, and the license file are named using the following naming conventions, where *[serialNumber]* is the serial number of the robot:

- License request file:
  no.rocketfarm.urcap.palletmanager.[serialNumber].license_request
- License file:
  no.rocketfarm.urcap.palletmanager.[serialNumber].license

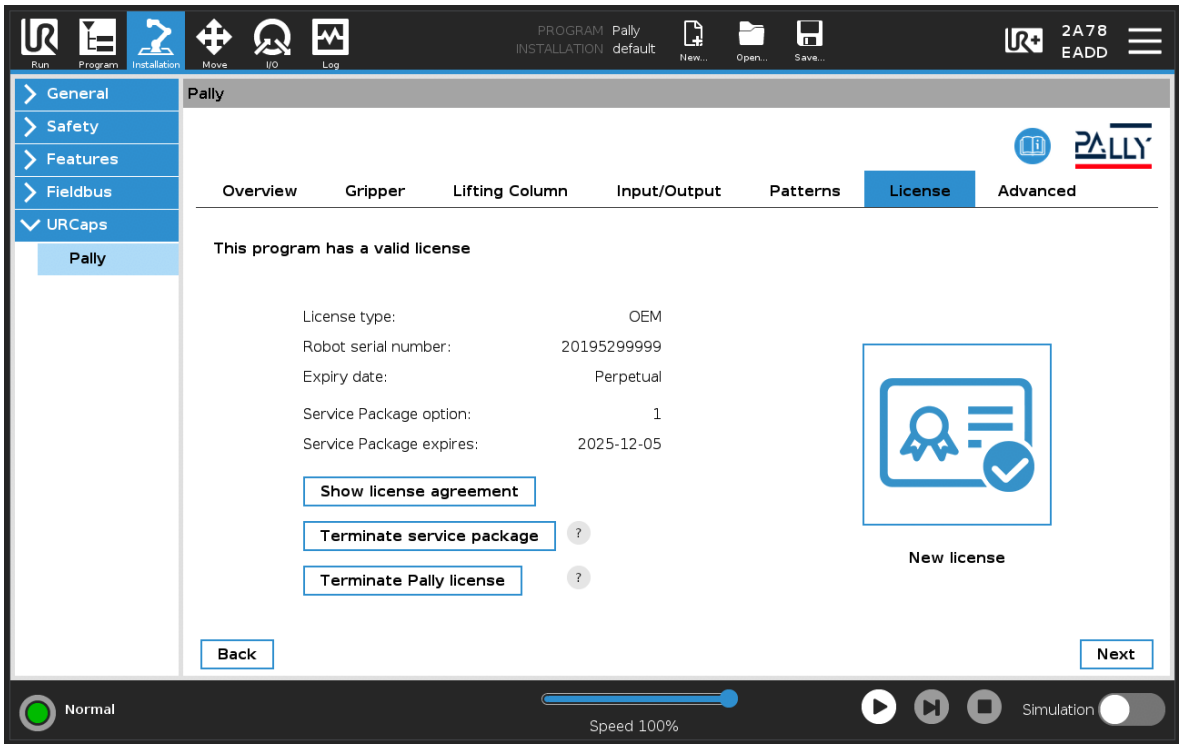


Figure 37: The view confirming the installed license is valid.

Some extra features like the Pally Operator Panel are limited to licenses with a Service and Support plan. The Service and Support plan status can be also checked on the License page.

6.1.2.2 - Terminating a Service and Support plan

If the Service and Support plan is no longer needed, insert a USB disk and press the "Terminate service package" button. The program will deactivate the Service and Support plan and generate an evidence file on the USB.

The following functions are not available without a valid Service and Support plan:

- Change gripper type and dimensions
- Change lifting column type and dimensions
- Upload new patterns, delete existing patterns, use patterns that are not 'verified'

- Change calibration points and calibration box dimensions
- Change path planning parameters
- Change operator language

The name of the evidence file has the following format:
no.rocketfarm.urcap.palletmanager.[serialNumber].status

6.1.2.3 - Terminating a license

If the license is no longer needed, insert a USB disk and press the "Terminate license" button. The program will deactivate the license and generate an evidence file on the USB.

The URCap can be safely uninstalled after the evidence file has been generated and sent to Rocketfarm.

6.1.3 - Gripper

In this tab you can choose the gripper manufacturer and model, dimensions and weight, the IO channels connected to the gripper, and the mounting offset.
First choose the manufacturer of your gripper:

- Piab
- Schmalz
- UniGripper
- other

If you don't find the manufacturer in the drop-down list, select 'other'.

Choose the gripper model that is mounted on the robot. The dimensions, weight, TCP, and IO settings will be set automatically for supported gripper models.

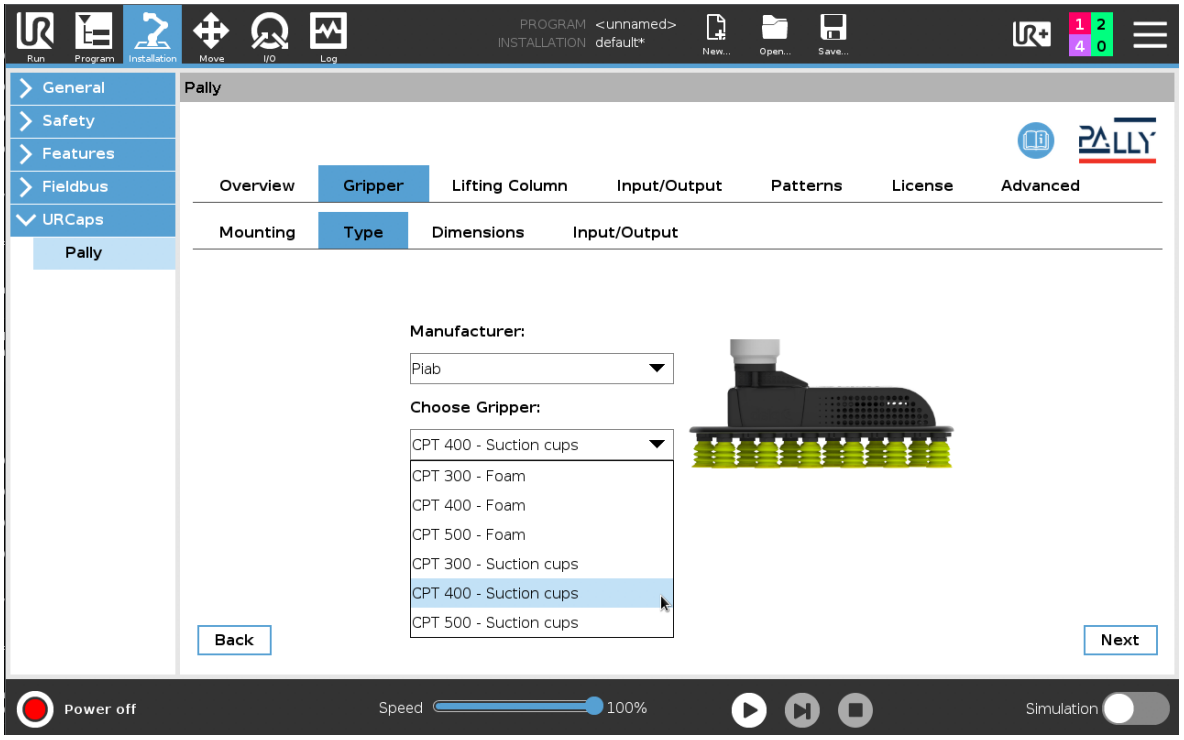


Figure 38: Choose the gripper manufacturer and model

For specific gripper models, the dimensions, weight, TCP, and IO channels will be automatically preselected. Custom grippers require these values to be filled according to the gripper specifications. Please note: gripper length is the dimension which is parallel to the conveyor direction.

Some gripper models e.g. the Piab CPT supports more than one offset/mounting option. Select the correct offset that corresponds to the actual mounting. A TCP called "PallyTCP" will be automatically created; make sure to set it as default TCP in Installation General TCP.

If you use a tool changer or mount your gripper using a custom made offset plate, select the option "Something else (set your TCP)" and configure the robot TCP in Installation General TCP.

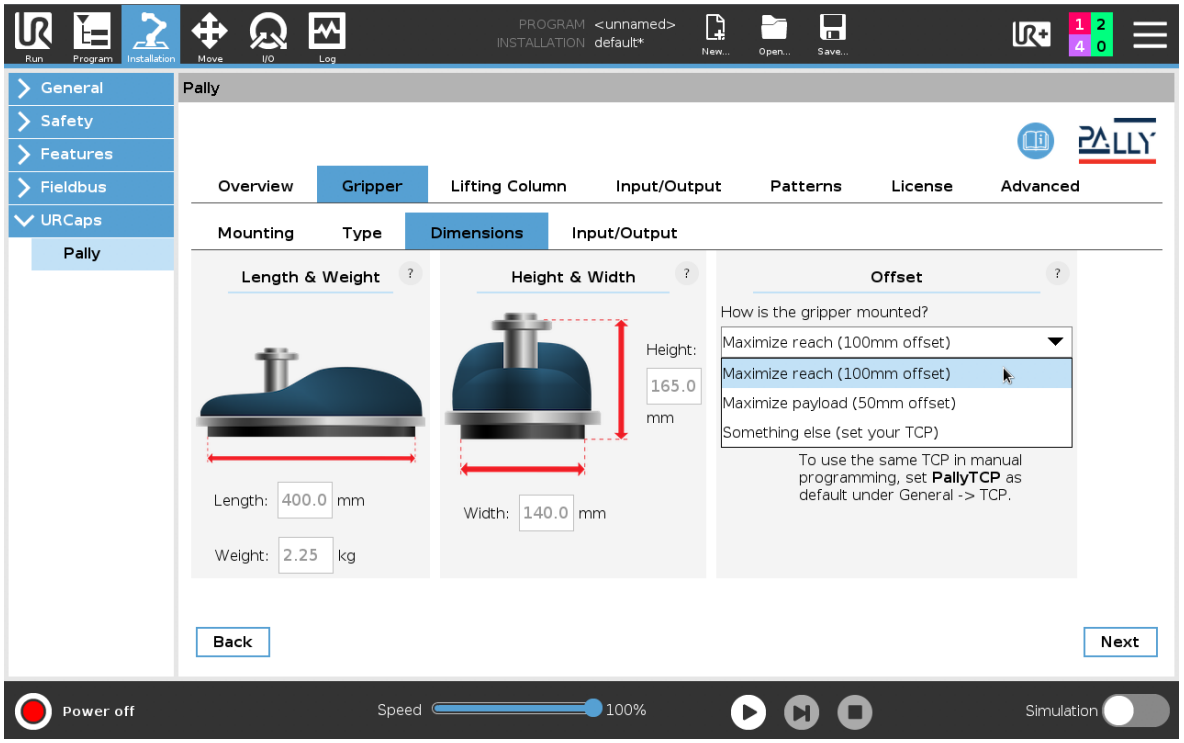


Figure 39: Gripper dimensions and offset

The corresponding IO channels are automatically preselected for grippers that are connected to the tool IO, e.g. the Schmalz FXCB models. Other grippers require the IO channels to be chosen manually.

For grippers that support compressed air blow, select the "blow on" signal. By using this signal, compressed air will be used to blow dust off the box before picking, and it also makes the release process faster.

The 'Vacuum lost' signal can be used to detect false picks and lost boxes. The program evaluates the signal when picking the box and also before dropping the box on the pallet. To avoid empty holes in the pallet pattern, the last position is automatically repeated when loss of vacuum is detected. Please note that your gripper may require special configuration or extra hardware components to use this feature.

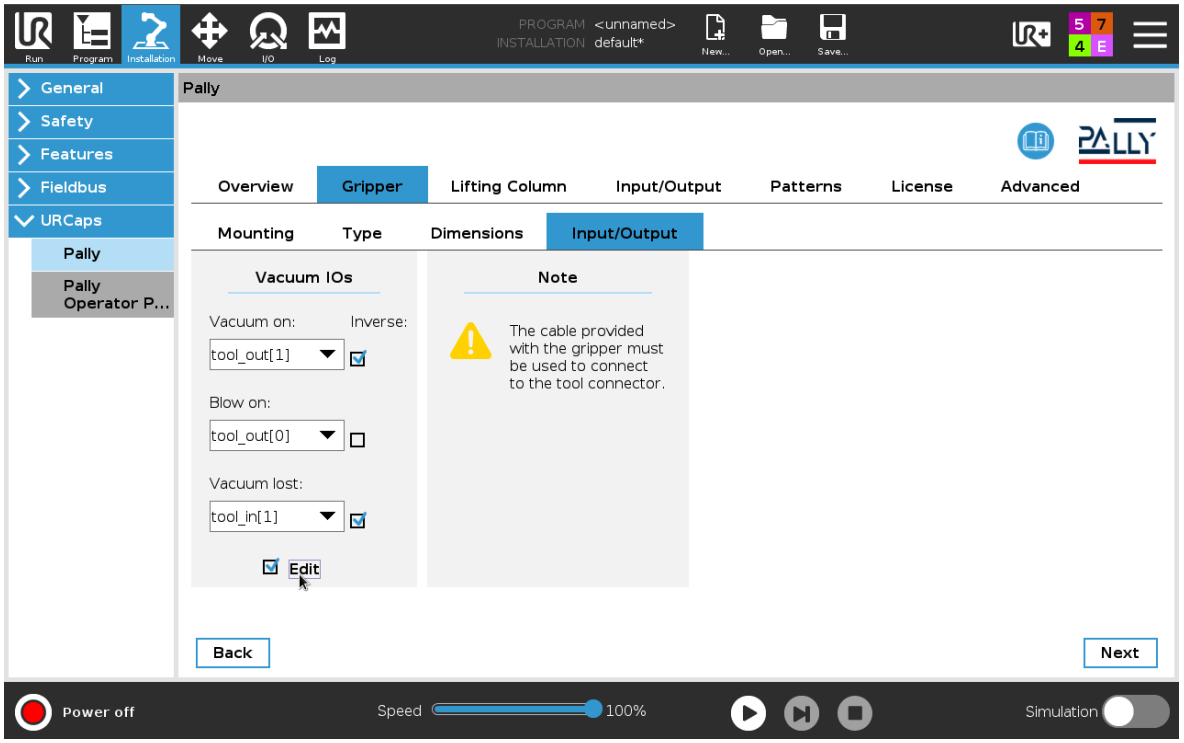


Figure 40: Select the gripper IO channels for vacuum, blow, and feedback.

If *Custom gripper* is selected, no IO channels can be set in the Pally installation node. Control the gripper using callbacks, see 6.3 - Callbacks.

6.1.3.1 - Supported gripper models

Piab CPT - Foam and Suction cups
Choose this option if a Piab CPT gripper has been mounted on the robot. The weight and dimensions of the gripper are selected automatically for the chosen model. Select the suction and blow signals according to the hardware connections.
Piab CPT grippers can be mounted with 50 or 100 mm offset by rotating the mounting bracket. Select how the gripper is attached: choose between 100 mm and 50 mm offsets and Pally will automatically create a TCP for the corresponding selection. If your gripper is mounted on a tool-changer or a custom made offset bracket, select 'Something else - Set your TCP'.

Note: Make sure the TCP corresponds to the actual mounting offset.

Schmalz FXCB - Foam and Schmalz FXCB - Suction cups
Choose this option if a Schmalz FXCB Foam gripper or a Schmalz FXCB Suction cup gripper has been mounted on the robot. Once the type of gripper is set to one of these

Schmalz grippers, the IO channels, weight, and dimensions are predefined. The grip and release functions are controlled by the Pally URCap via the tool digital IO channels.

Note: Make sure the Tool IO voltage is set to 24V.

Note: In order to get reliable feedback from the vacuum sensor, you need to configure the SP2 and rP2 values according to the VSi Vacuum/pressure switch Brief operating instructions. If you can't use or don't want to use the vacuum feedback in, click on 'Edit' and select 'Always low'.

UniGripper Co/Light Regular
Choose this option for standard mounted Unigripper Co/Light gripper only, as weight and dimensions are predefined. The grip and release functions are controlled by the Pally URCap via the digital IO channels configured here.

6.1.3.2 - Integrating other grippers

Custom Vacuum Gripper
Choose this option for other vacuum grippers that can be controlled by simple digital IO signals. Weight and dimensions of the gripper must be manually entered using this option. The grip and release functions are controlled by the Pally URCap via the digital IO channels configured here.

Custom Gripper
Choose this option if the gripper has its own URCap to control grip and release functions. Weight and dimensions of the gripper must be manually entered using this option. See 6.3 - Callbacks for further details on inserting custom commands into the program sequence.

Note: For grippers in this category, it might be necessary to manually configure TCP (position, orientation, payload, center of gravity) according to the technical specifications of your gripper. Consult your gripper manufacturer for further details.

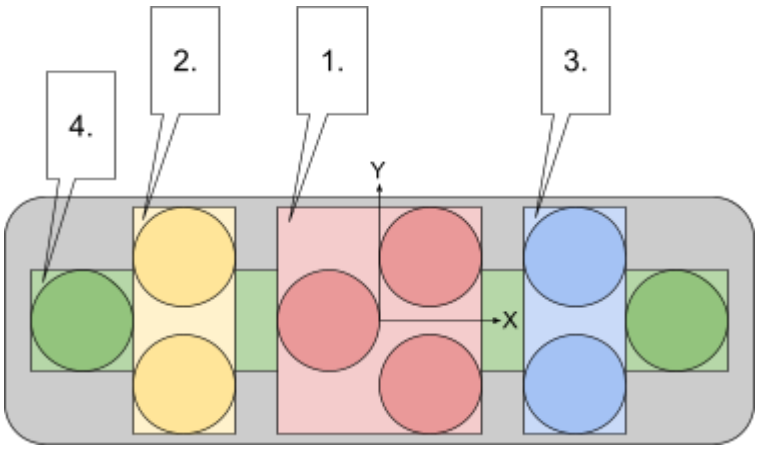
Import from gripper.json
Choose this option only if you have a special gripper with one or more vacuum channels and a file in the robot *home folder* called "gripper.json" that contains the geometry and the corresponding IO settings for the gripper.
In order to have this functionality working, the position and the dimensions of the bounding rectangle should be defined for each vacuum zone as well as the digital output channels that control the vacuum valves and optionally the compressed air used at release. The file has the following structure:

- name: name of the gripper
- description: a short description of the gripper

- picture: an optional picture in PNG format, encoded as a base64 string
- dimensions:
  - width, length, height: total dimensions of the entire gripper
  - weight: weight of the gripper including mounting bracket and screws
  - foamHeight: the height of the foam / suction cups that can be pressed down
- properties:
  - edgeAlignment: if true, the box edges will be aligned to the gripper zone edges, otherwise the gripper zone(s) will be centered above the box center
  - coveragePercent: defines how many percents of the vacuum zone (bounding rectangle) area must be covered in order to use a specific zone
- tcp: the tool center point (TCP) of the gripper as [x, y, z, rx, ry, rz]
- cog: the center of gravity (COG) of the empty gripper as [x, y, z]
- zones: definition of each individually controllable vacuum area
  - id: unique identifier (number) of the vacuum area, starting from 1
  - x, y: center point of the bounding rectangle relative to the gripper center
  - width, length: size of the bounding rectangle
  - grip: digital IO signal that controls the vacuum
    - type: standard, configurable, tool (robot IO type)
    - channel: the digital output channel (0-1 for tool 0-7 otherwise)
    - inverse: true if high level signal means no vacuum
  - release: digital signal that controls the compressed air (optional)
- configurations: list of valid combinations of vacuum areas that can be used together

(Lengths are specified in millimeters, weight in kilograms.)

The following diagram is an example of a 4-channel suction-cup gripper with its corresponding gripper.json file. The different colors indicate different groups of suction cups that can be controlled together.




```
{
  "name": "4-channel gripper",
  "description": "test gripper for multiple product sizes",
  "dimensions": {
    "width": 628,
```

```
    "length": 252,
    "height": 202,
    "weight": 1.9,
    "foamHeight": 15
  },
  "properties": {
    "coveragePercent": 51,
    "edgeAlignment": false
  },
  "tcp": [0, 0, 110, 0, 0, 90],
  "cog": [0, 0, 80],
  "zones": [
    {
      "id": 1,
      "x": 0,
      "y": 0,
      "width": 134,
      "length": 201,
      "grip": {
        "type": "standard",
        "channel": 0,
        "inverse": false
      },
      "release": {
        "type": "standard",
        "channel": 4
      }
    },
    {
      "id": 2,
      "x": -163,
      "y": 0,
      "width": 67,
      "length": 201,
      "grip": {
        "type": "standard",
        "channel": 1
      },
      "release": {
        "type": "standard",
        "channel": 5
      }
    },
    {
      "id": 3,
      "x": 163,
      "y": 0,
      "width": 67,
      "length": 201,
```

```
      "grip": {
        "type": "standard",
        "channel": 2
      },
      "release": {
        "type": "standard",
        "channel": 6
      }
    },
    {
      "id": 4,
      "x": 0,
      "y": 0,
      "width": 527,
      "length": 67,
      "grip": {
        "type": "standard",
        "channel": 3
      },
      "release": {
        "type": "standard",
        "channel": 7
      }
    }
  ],
  "configurations": [
    [1,2,3,4], [4], [1,2,3], [1,2], [2,3], [1], [2], [3]
  ]
}
```

Figure 41: suction-cup gripper with 4 channels and the corresponding gripper.json file

---

Note: TCP settings are optional and may not be included in older gripper.json files. If your gripper is mounted with a position/rotation offset, make sure that the offset is properly set according to the specification in 5.1.3 - Attach the gripper properly

---

6.1.4 - Lifting column

Lifting columns can be selected on this configuration page. The following options are available:

No lifting column
Select this option when no lifting column is in use.

Ewellix LiftKit

Select this option when the robot is mounted on top of an Ewellix LiftKit lifting column. Select "Use dynamic positioning" if your project requires recalculating the lifting column position for each pick and place movement. Enter the maximum stroke in millimeters. Pally will dynamically calculate the best possible lifting column position for each layer, depending on the product dimensions.

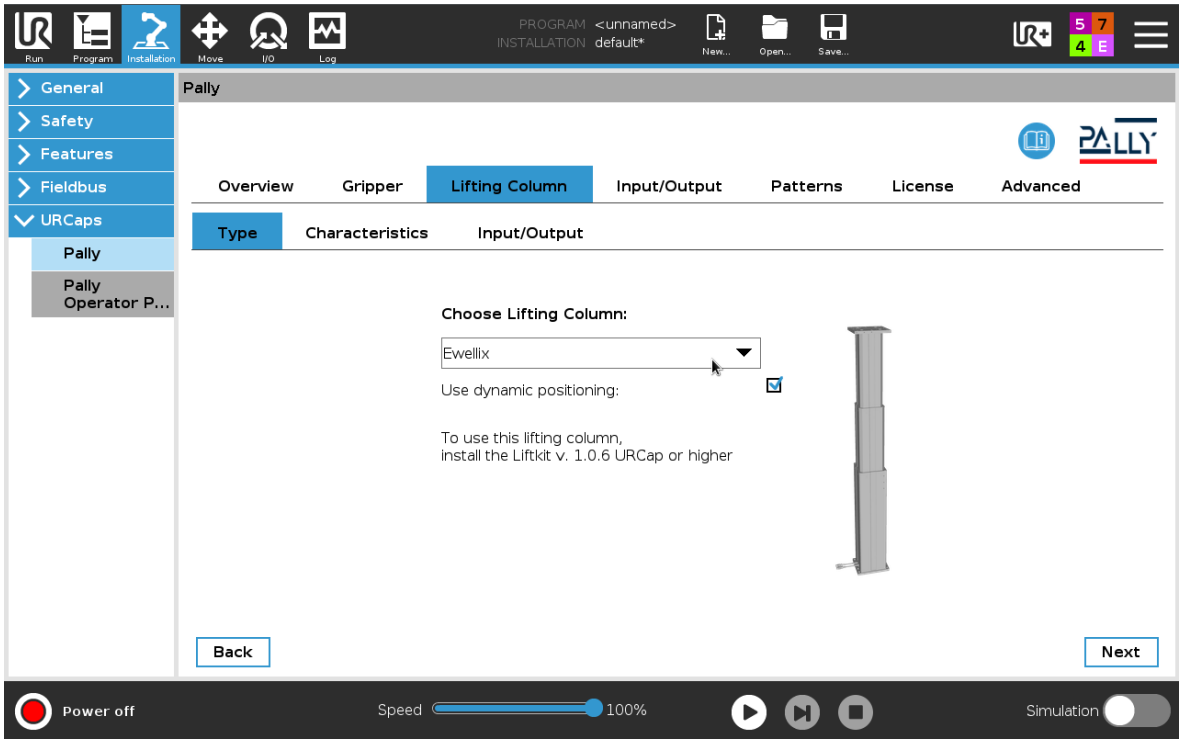


Figure 42: Ewellix LiftKit is natively supported

Note: Make sure Ewellix LiftKit URCap version 1.1.2 or newer is installed and configured.

Vention V1

Select this option for lifting columns that are built with the first generation Vention MachineMotion controllers. Select "Use dynamic positioning" if your project requires recalculating the lifting column position for each pick and place movement. Enter the maximum stroke in millimeters, and the controller name and drive number as they are configured in the "Vention Robot Cell" URCap.

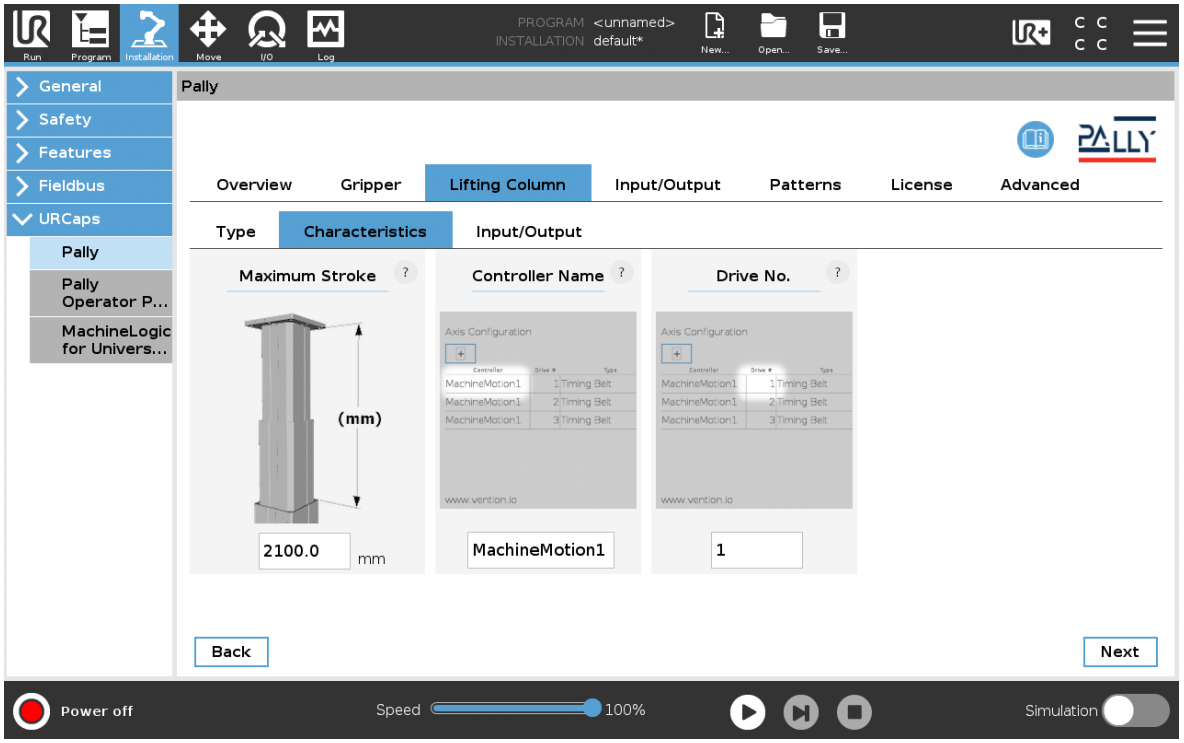


Figure 43: Vention lifting columns require additional settings to work properly

---

Note: Make sure the URCap "Vention Robot Cell" v2.0.1 or newer is installed and configured. This URCap is provided by Vention.

Vention V2

Select this option for lifting columns that are built with Vention MachineMotion V2 controllers. Select "Use dynamic positioning" if your project requires recalculating the lifting column position for each pick and place movement. Enter the controller name and drive number as they are configured in the "MachineLogic for Universal Robots" URCap.

Note: Make sure the URCap "MachineLogic for Universal Robots" v3.1.5 or newer is installed and configured. This URCap is provided by Vention.

Note: The encoder zero position can be set to any position in the MachineLogic URCap. Make sure to select a zero position that is suitable for the whole calibration process. Set the minimum and maximum stroke parameters according to how much the lifting column can go below- and above the zero position, see the rf_min_stroke and rf_max_stroke parameters.

Note: The positive move direction can be set in the MachineLogic URCap. Make sure to configure the directions so that position values greater than zero will move the lifting column above the zero position.

Linak Elevate (Easy, Pro, Modbus)
Select one of these options for the corresponding Linak Elevate lifting column. Refer to the lifting column user manual for further details on the configuration settings.
Make sure the Elevate URCap provided by Linak is installed and configured correctly.

Note: Elevate Easy lifting columns can only have two valid positions: fully stretched and fully retracted.

Custom Lifting Column
Select this option for lifting columns that can be controlled by 24V digital signals to move up and down. Pally will not calculate any intermediate positions between retracted length and maximum stroke. Following parameters are available:

Control signals	Digital outputs that will control the movement of the lifting column.
End-switches	Digital inputs that are high when the lifting column has reached one of its end positions.
Motion feedback	Digital input that tells whether the lifting column is in motion.
Max. duration	Time limit in seconds allowed for the lifting column to move from one end position to another.

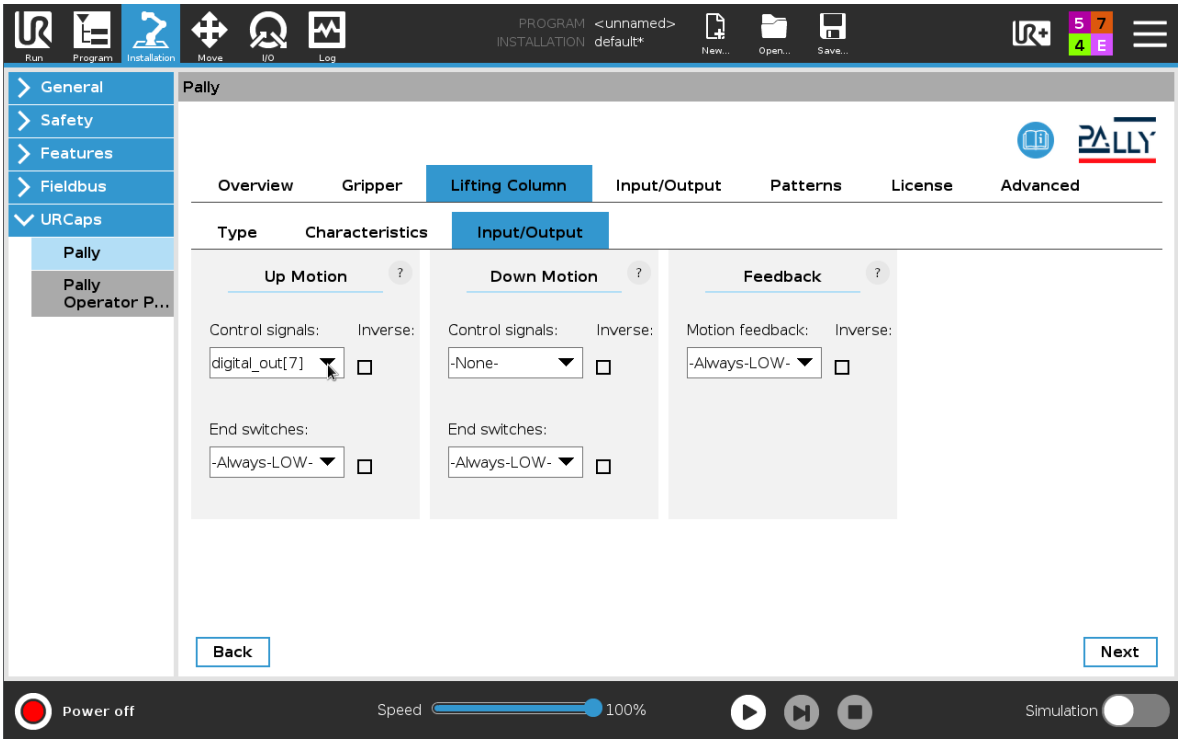


Figure 44: Lifting column with digital I/O signals

Note: It is highly recommended to use a lifting column with feedback from end-switches and current motion state. Without feedback signals, the predicted state of the lifting column cannot be verified and may lead to unexpected error messages or malfunction in Pally.

Note: Custom lifting columns can only have two valid positions: fully stretched and fully retracted.

6.1.5 - Input/Output

The assignment of input/output channels to each hardware unit can be configured here.

Inverted signals are supported; where an inverted signal means that the signal goes from high to low when a specific event occurs (e.g product present). On the contrary, the default signal (i.e not inverted signal) means the signal goes from low to high when a specific event occurs. To use an inverted signal, simply click on the ‘Inverse’ checkbox next to the dropdown for the signal value.

For input signals it is also possible to select the "Always LOW" and "Always HIGH" options. In this case, no physical input channel is being evaluated, and the result is constantly low or constantly high, respectively. For output signals it is possible to select "None". In this case the signal won't appear on any physical output channel.

It is very important to select different channels for each unit. For instance, do not use digital_input[0] for pallet presence AND product presence sensor, as this will result in unexpected behavior.

Refresh IO
Push this button to reload the IO channel names after making changes on the General I/O Setup screen to show the actual names of each IO channel.

6.1.5.1 Using MODBUS or General Purpose Boolean Registers

In addition to the standard digital I/O and configurable digital I/O, it is now possible to select and use the general purpose registers and external Modbus signals with Pally.

Modbus Registers
In order to use MODBUS registers in Pally, a MODBUS connection has to be set up in the ‘MODBUS client IO Setup’ in the Polyscope installation node.

For the specified MODBUS Unit, add a ‘Digital Input’ or a ‘Digital Output’ signal. It is only possible to use the *digital* input or output signals from a MODBUS unit with Pally, not the *register* input or outputs.

Note: The name of the signal has to be kept equal to the default name given by Polyscope, such that the format of the signal name is: MODBUS_N where N is a number.

For this first version of supporting these Modbus IOs in Pally, the default signal names for the MODBUS signal need to remain unchanged.

General Purpose Boolean Registers
In order to use the General Purpose Boolean registers in Pally, the specified general purpose boolean IOs that will be used have to be renamed in Polyscope. This is a convention that is specified by Polyscope, and can be done in the ‘I/O Setup’ tab in the installation node.

Note: UR only allows using the first 64 General Purpose IOs with a Fieldbus/PLC, the rest of the General Purpose IOs are reserved for external RTDE clients. So, to add a General Purpose Boolean input/output, rename a GP IO in the range: GP_bool_in[0] - GP_bool_in[63] / GP_bool_out[0] - GP_bool_out[63]

Note: Out of all the General Purpose IOs (bool, float and int) that UR supports, Pally will only work with the General Purpose *Boolean* IOs.

Once the general purpose boolean IOs are renamed, navigate to the Pally installation node and go to the ‘Advanced’-tab → ‘System’-tab, and press the ‘Refresh’-button next to the ‘Refresh IO-lists’-label. It will now be possible to select the general purpose IOs in the configuration of the Pally installation node.

6.1.5.2 Input/Outputs in Pally

Confirm pallet signal
These input signals shall be connected to physical push buttons where the operator can confirm that the empty pallet is present and ready for palletizing. After a pallet has been finished, the program will not start palletizing the same pallet position again until the respective signal is received for at least 0.2 seconds.
Empty pallets can be also confirmed by using the Pally Operator Panel. In this case the physical signals can be omitted.

No pallet signal
These output signals can be connected to physical light sources, for example, to illuminate the pallet present buttons. The light will flash with 1Hz while the program is waiting for the pallet present button to be pushed.
When using the Pally Operator Panel the pallet state is displayed on the teach pendant. In this case the physical signals can be omitted.

1st product sensor
These input signals shall be connected to sensors that are mounted at the end of the conveyor belt, indicating that a product is ready to be picked up. At least one product sensor is mandatory.

2nd product sensor
These input signals can be connected to sensors that are mounted one box behind the 1st product sensor, indicating that the robot can pick 2 products (where available).

Add new sensor
Push this button to add more sensors. Maximum 8 product sensors can be created for one production line.

Remove selected sensor
Push this button to remove the selected sensor. The first product sensor cannot be removed.

Priority sensor
These input signals can be connected to sensors that are mounted 3 or more boxes behind the 1st product sensor, indicating that the specific conveyor has too many products to be

palletized. Only effective when 2 conveyors are configured. The program will prefer the pickup position with priority signal on.

LED-tower
These output signals can be connected to light sources that indicate the status of the palletizer robot as follows.

Red	Program stop (emergency stop, normal program stop)
Yellow	Only one pallet is present, the robot will stop when the current pallet is full
Green	Both pallets are present, the robot will continue with the next when current pallet is full

Do not use the Green output in projects with only one pallet position, as it will never be ON. In this case use the Yellow output to indicate normal operation.

Note: It is recommended to give meaningful names to the input/output channels to represent their physical meaning in the I/O Setup menu in the installation tab.

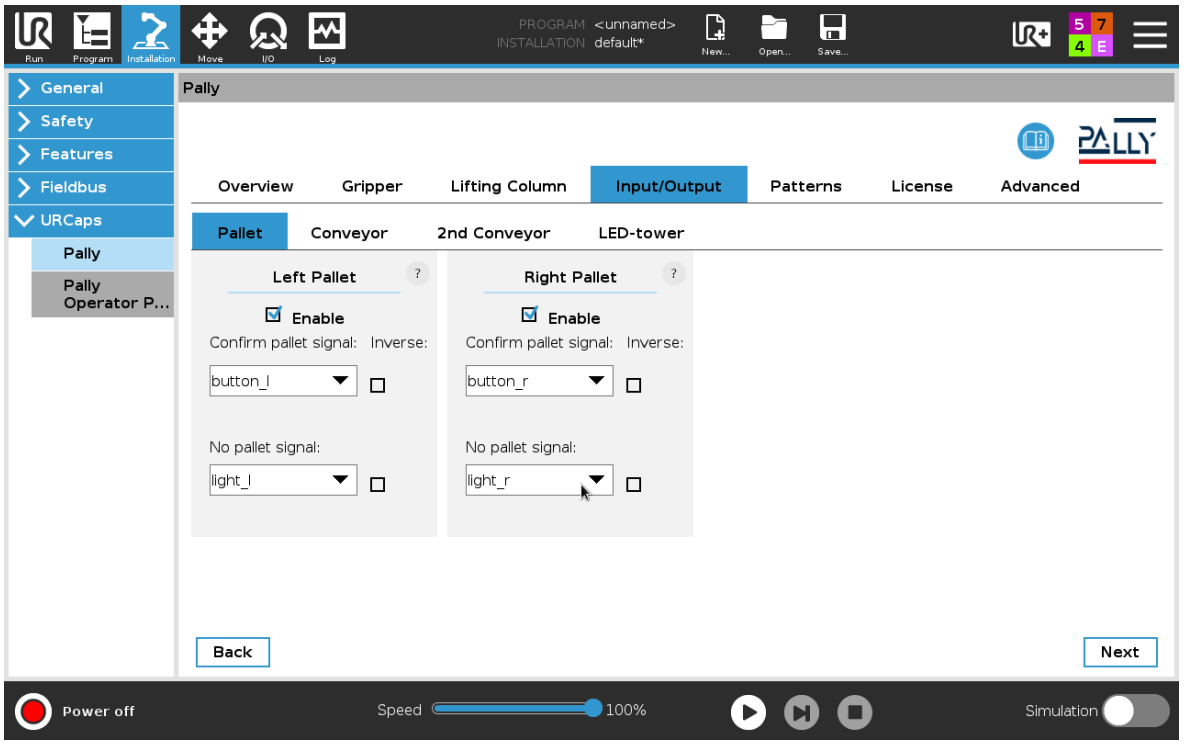


Figure 45: Go through each tab in the 'Input/Output'-tab to configure the inputs and outputs that go with the desired set up.

6.1.6 - Patterns

This configuration screen contains patterns specific commands.

Patterns storage directory
This field is used to change the path to the directory that is used to store the JSON-files that contain the palletizing patterns.

In order to edit this field, the daemon has to be stopped. Therefore, if this field is disabled, then navigate to the ‘System’-tab (or click the ‘Next’-button in the current tab) and press the ‘Stop daemon’-button.

Patterns in the storage directory
This table shows all the patterns that are uploaded to the patterns storage directory on the robot. It is possible to delete patterns from the robot via the ‘Delete pattern’-button on the bottom of the table.

Automatic upload of patterns
This checkbox is used to allow for automatic uploading of patterns from a USB-stick onto the robot. The checkbox is *not* checked by default. By checking it, all JSON-files in the root folder of the most recently plugged in USB-stick will be uploaded to the patterns storage directory folder on the robot.

Manually upload patterns from USB
This button can be used to upload patterns from a plugged in USB-stick to the patterns storage directory on the robot. It is only possible to use this feature if the ‘Automatic upload of patterns’ is disabled.

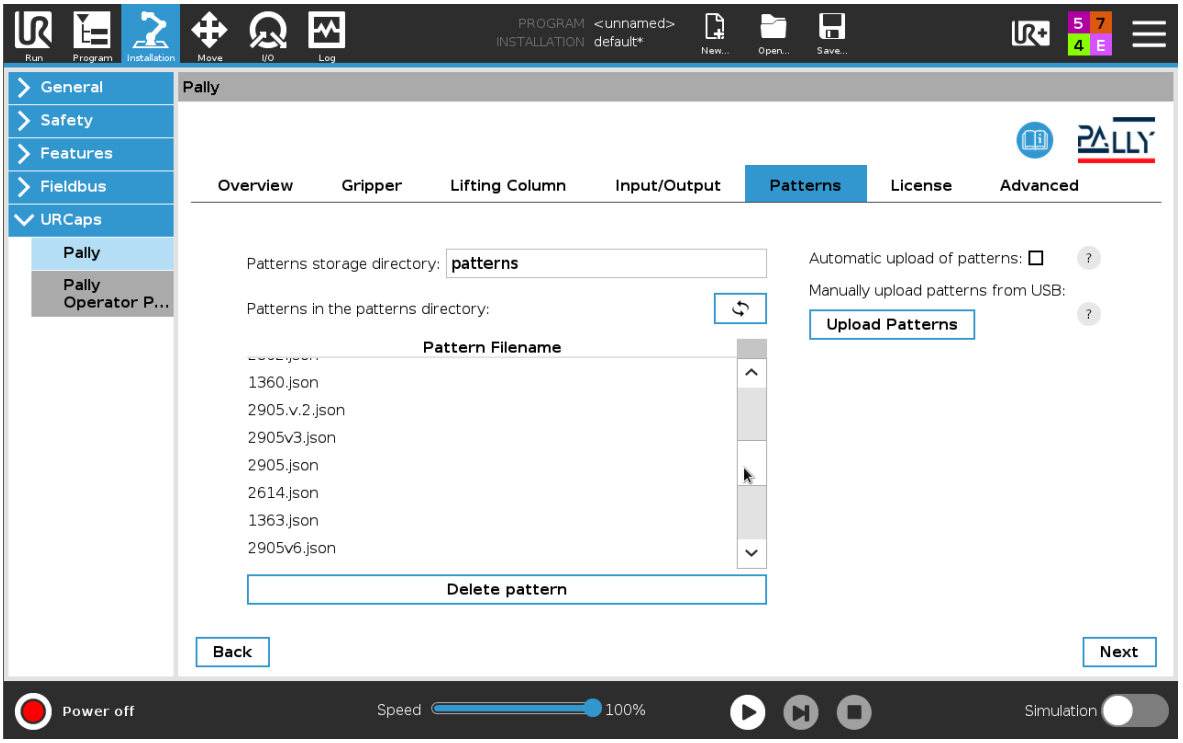


Figure 46: Pattern management.

6.1.7 - Advanced

For better visibility, the Advanced screen has been divided into the following tabs:

- Daemon
- Backup
- Other

This configuration screen is reserved for advanced features and options.

Daemon port number
Select the XMLRPC port number for communication with the PalletManager daemon process. The default value should not be changed unless it's in conflict with other installed URCaps.

Start daemon / Stop daemon / Daemon status
Start and stop the PalletManager daemon process and display its current running state.

Back up
By pressing this button, files from the following folders will be copied to the USB-stick that was plugged into the robot most recently:

- /root/.urcaps
    - From this folder, the installed URCap files are copied to the USB.
- /home/.no.rocketfarm.urcap.palletmanager

    - From this folder, Pally's license is copied to the USB.
- /programs
    - From this folder, the programs created on the robot *and* the patterns-folder are copied to the USB.
- /root/.urcontrol
    - Physical calibration data of this specific robot arm is copied to USB.

Back up Configuration
Press this button to make a copy of the current program and installation.

Operation mode
Select "Dual product" option to enable palletizing two different products from two pickup positions simultaneously, see 7.3.2 - Dual product mode, otherwise select "Palletizer" (default).

RoboDK simulation mode
Select this option to enable simulation with RoboDK software. This option is exclusively for developers. This option may be hidden in some setups.

Use alternative reach check
Select this option only if you have safety planes enabled and the robot stops frequently with false error messages saying positions are out of reach.

Reset to default
By pressing this button, the default values will be restored on the *Gripper, Lifting Column, Input/Output, Patterns,* and *Advanced* screens.

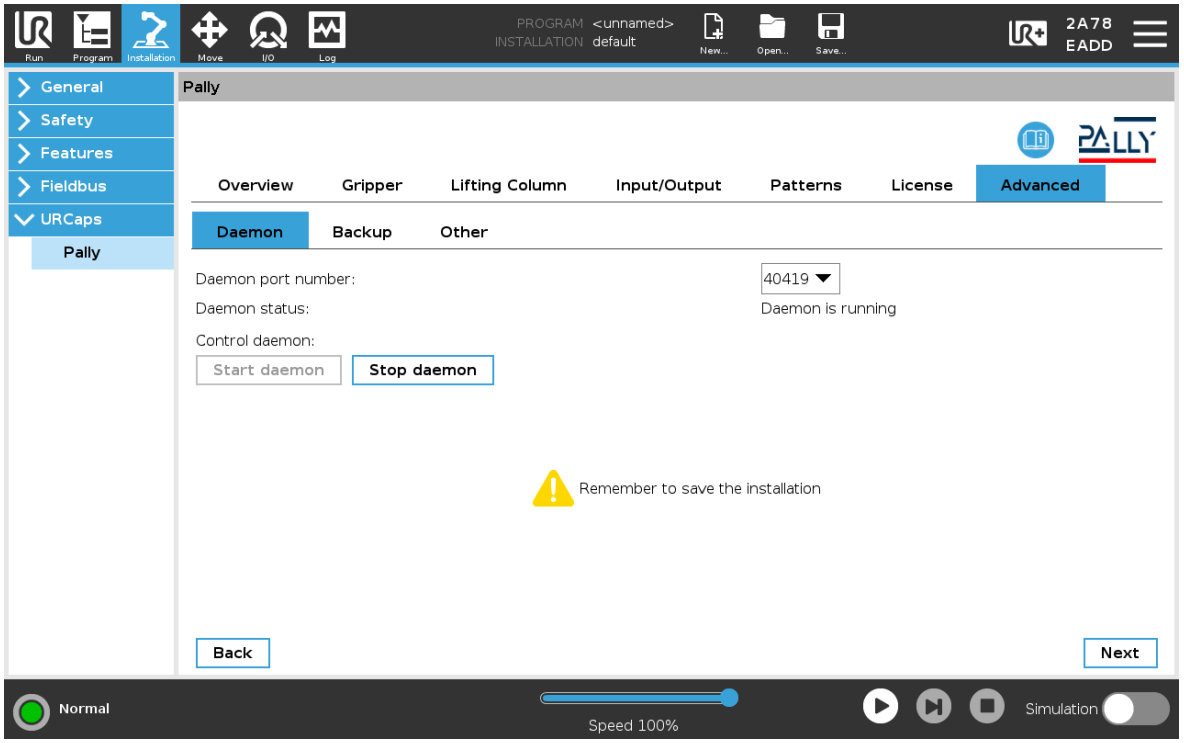


Figure 47: Advanced installation settings.

6.2 - Program settings

Create the palletizer main program first. Go to *Program Robot -> Empty Program -> Structure -> URCaps -> Pally*.

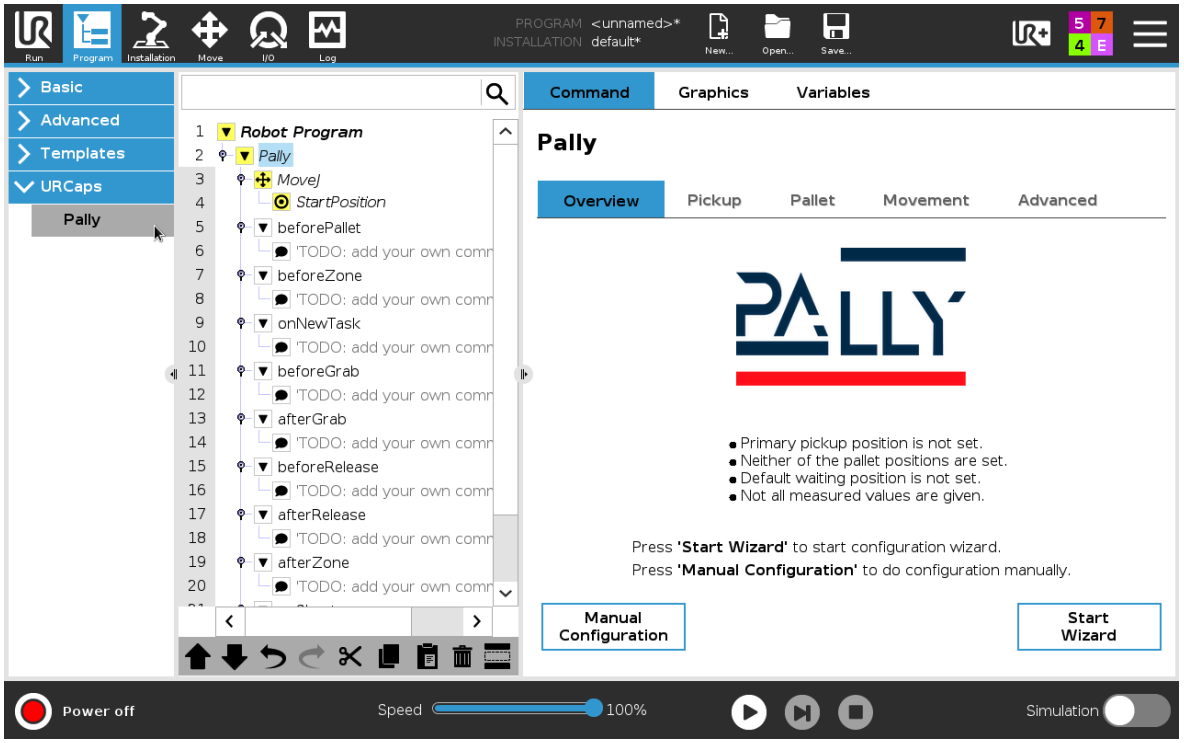


Figure 48: Add the palletizing program.

Press "Start Wizard" to navigate through all settings step by step, or use "Manual Configuration" to proceed to all configuration tabs.

6.2.1 - Pickup

Pickup position(s) should be calibrated by putting a box at the end of the conveyor and moving the gripper on the top of it. If possible, choose a calibration box with *at least* the width and length of the gripper.

Note: The lifting column has to be at its zero position during calibration.

Note: It is necessary to redo the calibration of the pickup position if the gripper dimensions on the *Installation* page are updated.

6.2.1.1 Calibration Box

Before calibrating the pickup position(s) enter the dimensions of the box that will be used for calibration, as shown in the figure below.

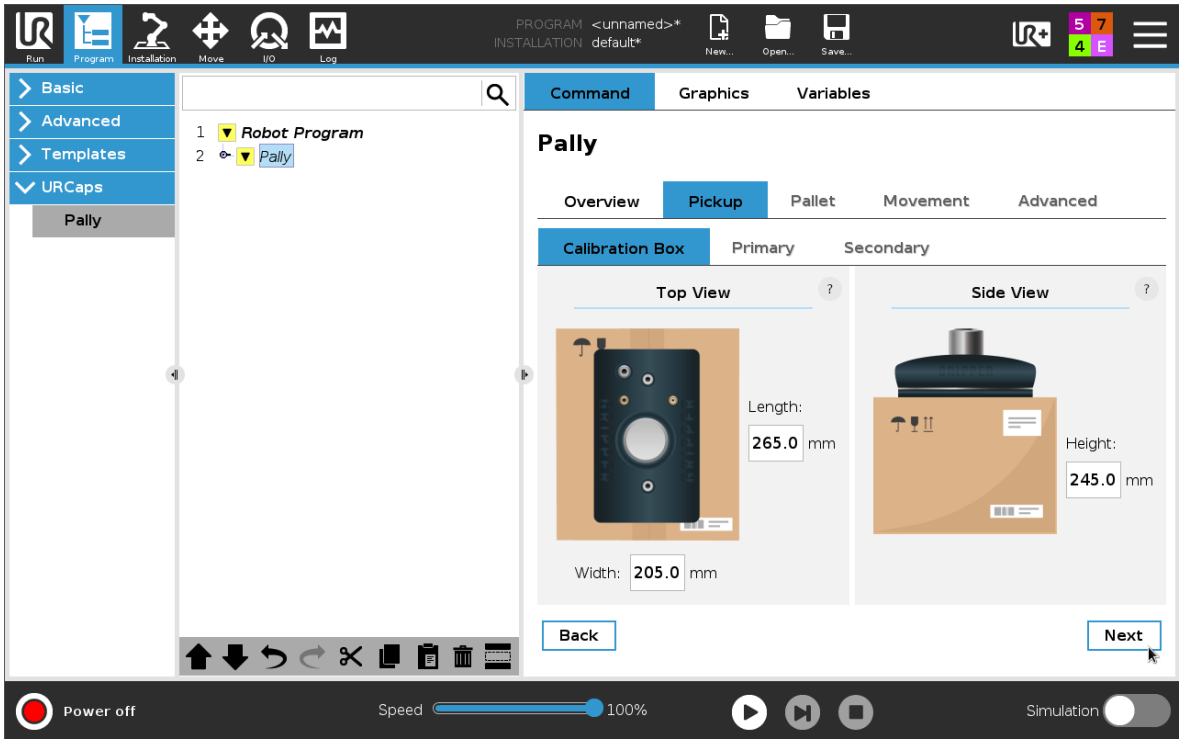


Figure 49: Calibration box

Note: The measurements must be accurate.

Align the gripper in the center of the calibration box, so that the orientation of the tool is perfectly perpendicular to the pickup position.

6.2.1.2 Primary Pickup

The primary pickup position is calibrated in the tab labeled ‘Primary’. When setting the robot’s position in the pickup position, make sure to use the calibration box that was specified in the ‘Calibration Box’-tab.

The production line name that is set is shown to the operator when the Pally program is running, it is used to distinguish between the two pickup positions/production lines when Pally is running in dual product mode (see section 7.3.2 Dual Product Mode). The value for the production line name defaults to ‘Line 1’ for the primary pickup, please rename it to a name that makes sense to the operators if dual product mode is enabled.

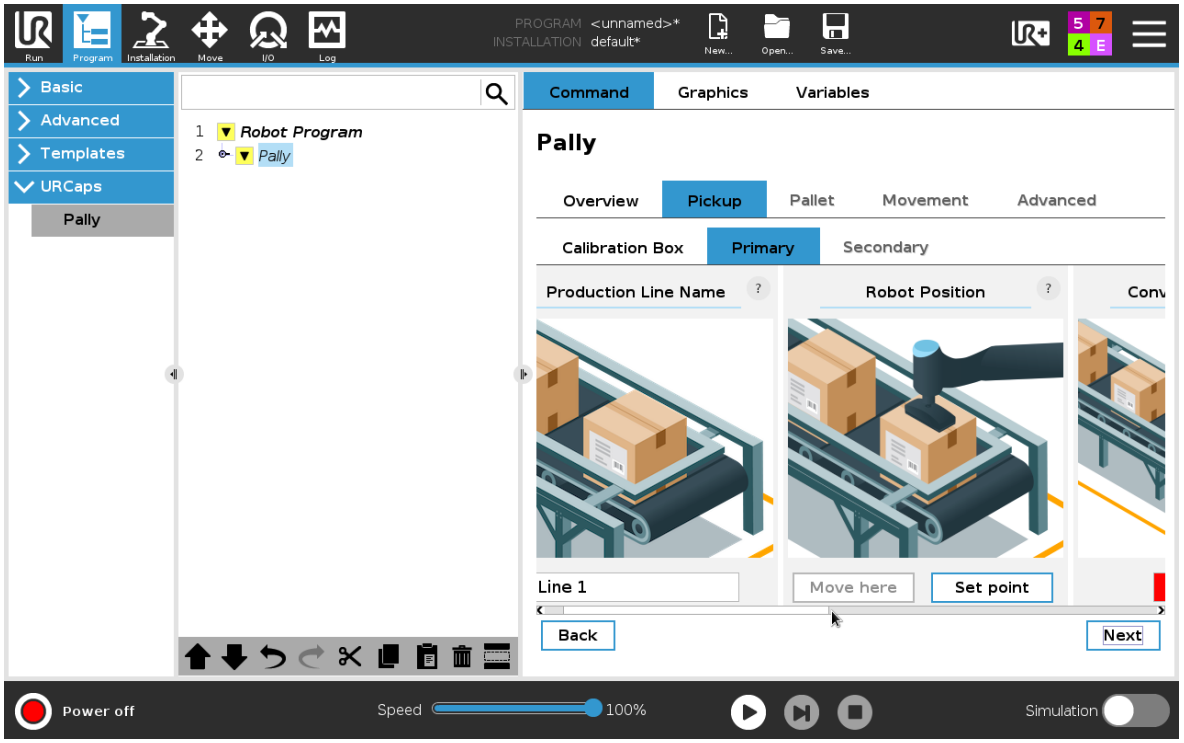


Figure 50: Naming the pickup position/production line

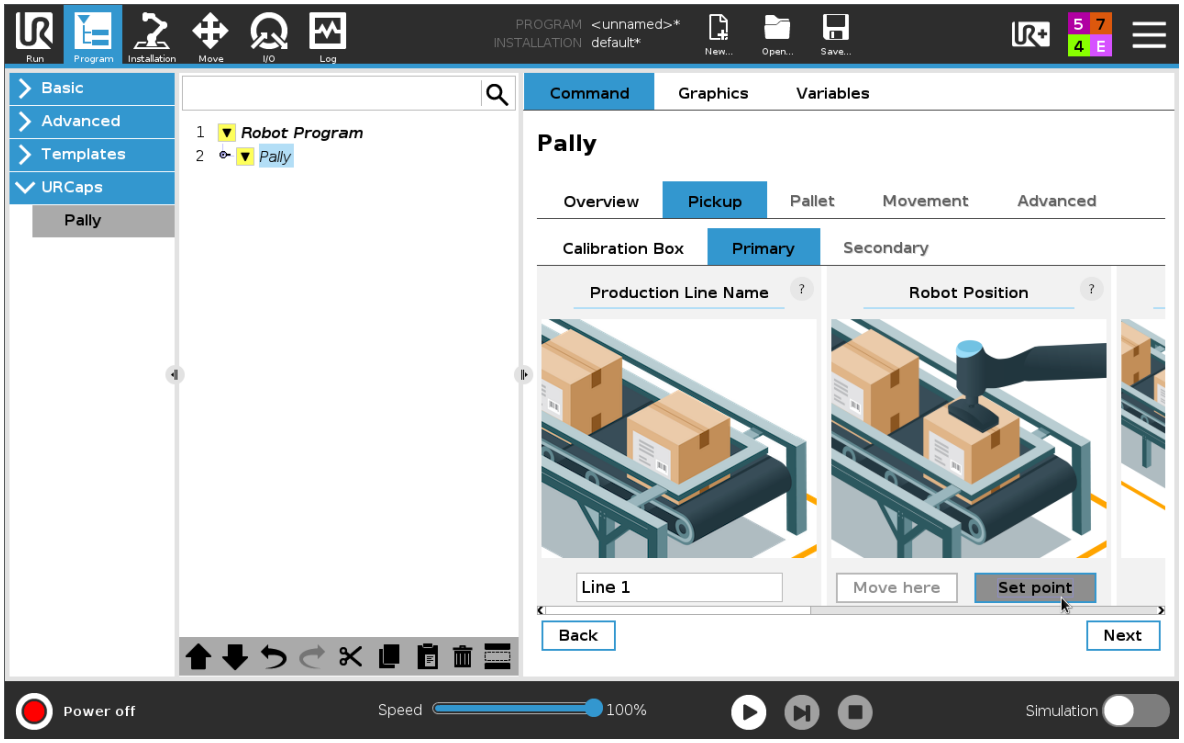


Figure 51: Calibrating the pickup position with a calibration box.

Note: The gripper should be centered on the top of the box. This might be difficult with a calibration box smaller than the gripper. Choose a larger box if possible.

Some grippers may have offset by design, or can be mounted with an offset.
The correct calibration of the pickup position with an offset-mounted gripper is shown below.

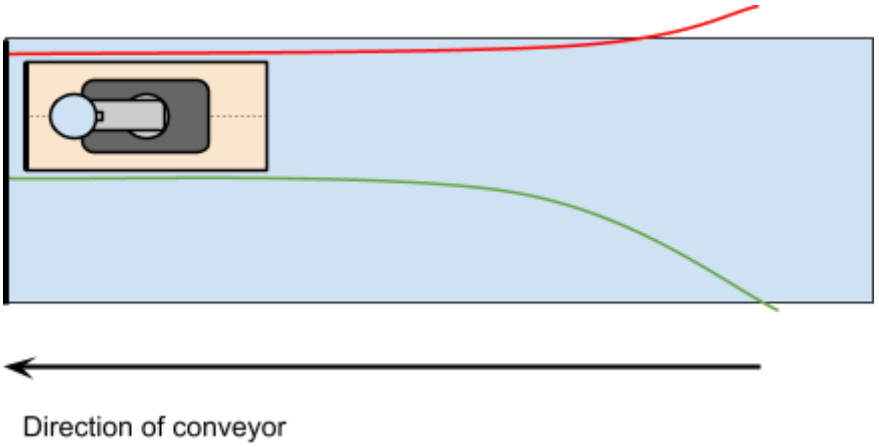


Figure 52: Calibration of the pickup position with an offset-mounted gripper

Enter the conveyor dimensions and select the fixed guide position as shown below.

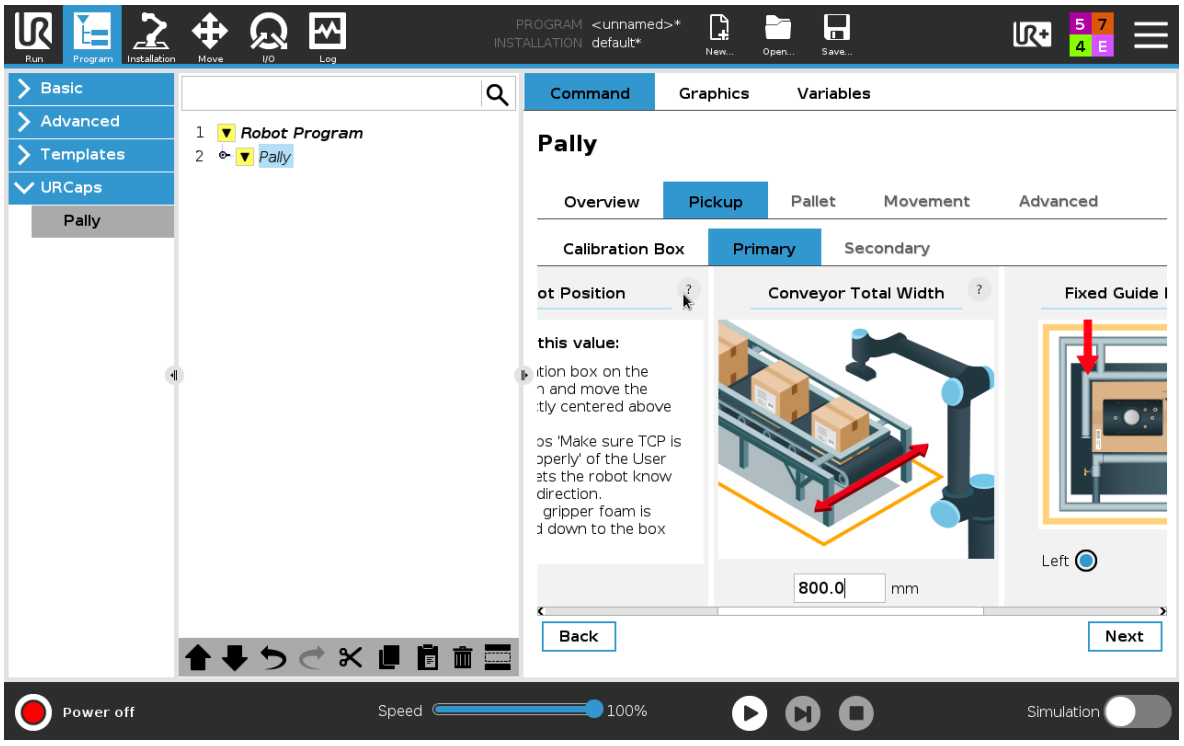


Figure 53: Total width of the conveyor at the pickup position.

Note: The fixed guide orientation is looking from the end of the conveyor and up the line.

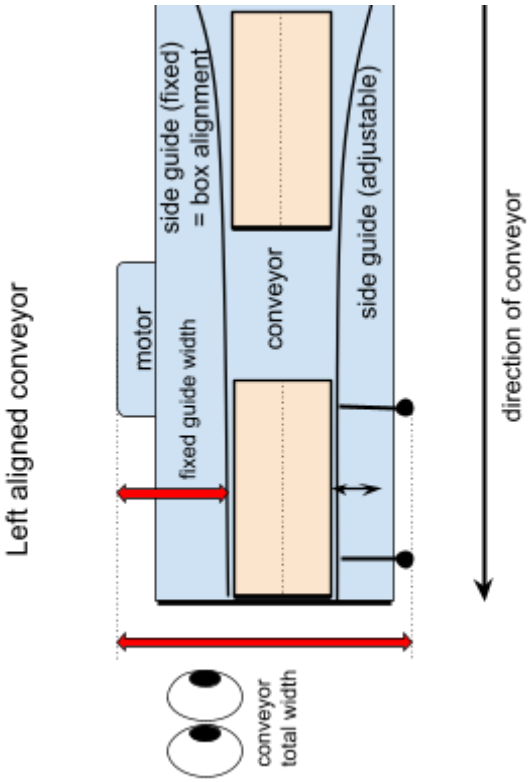


Figure 54: The fixed guide orientation is looking from the end of the conveyor and up the line. In this example the fixed guide position is *left*.

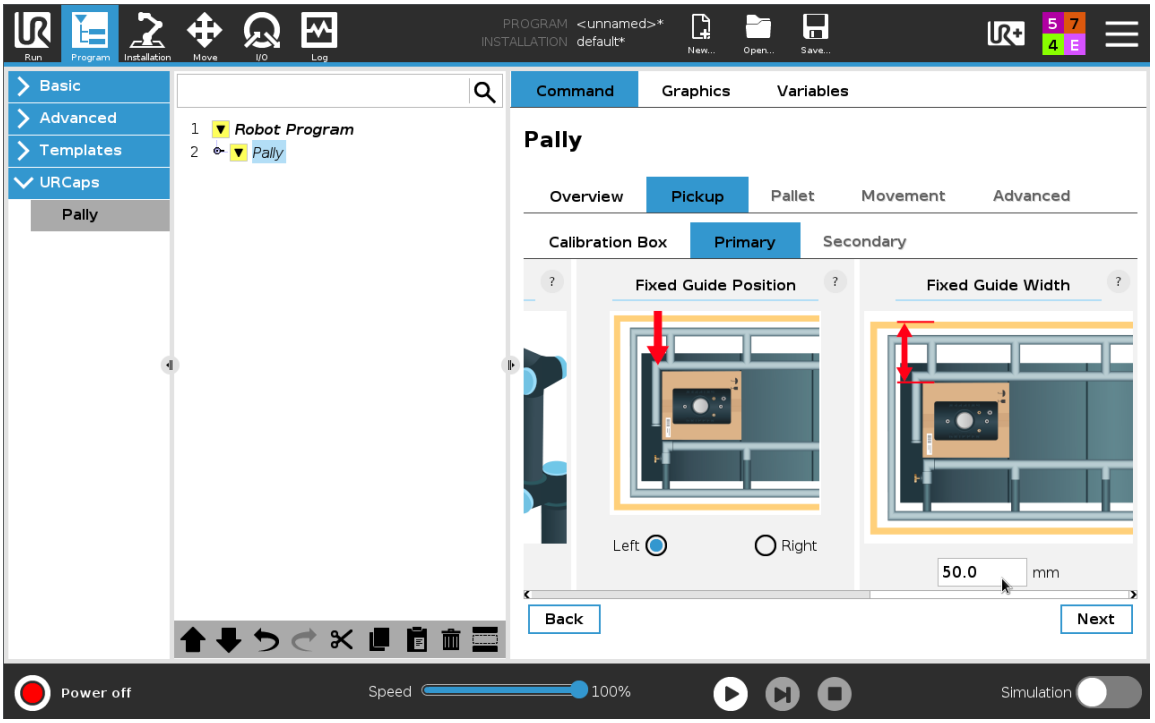


Figure 55: Configuration of the fixed side guide.

6.2.1.3 Secondary Pickup

In Pally, it is possible to use two pickup positions. To turn on this feature, check the ‘Enable’ checkbox (as shown below).

Note: If ‘Dual product mode’ is turned on (in Pally installation node), then the second pickup position *must* be set to run the program. It is possible to not use this production line when the program is running however, see section 7.3.2 Dual Product Mode for more.

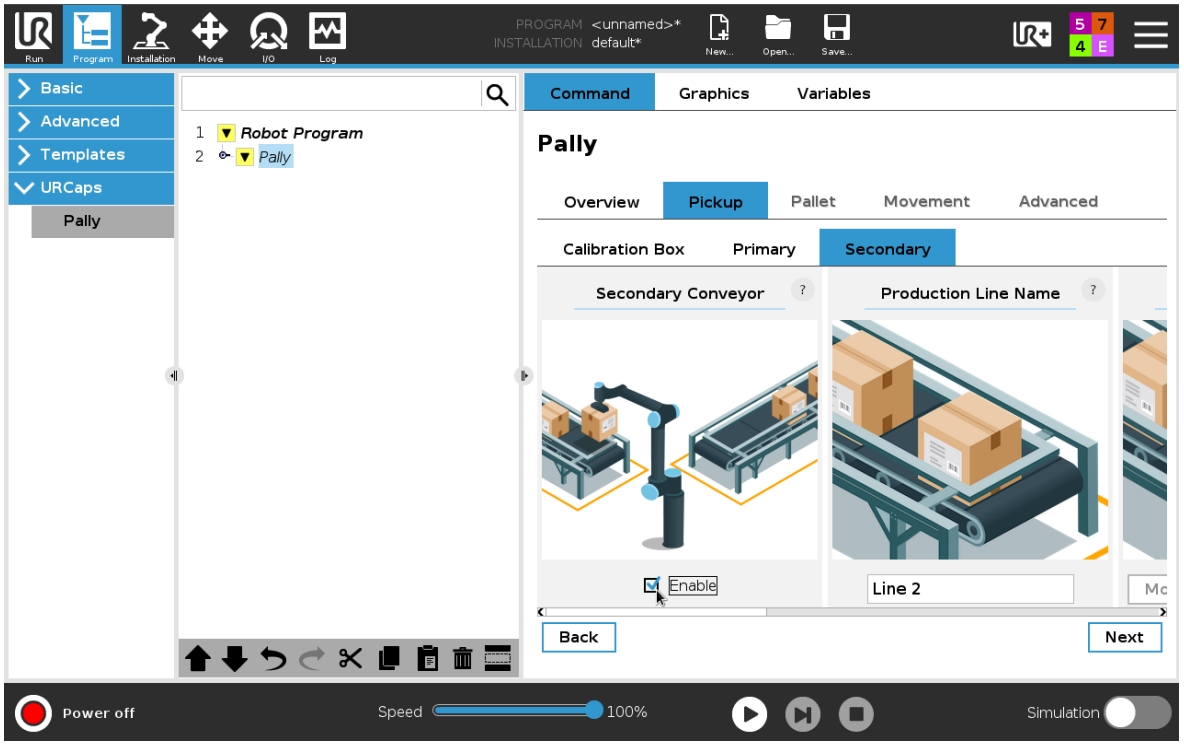


Figure 56: Enabling two pickup positions

As for the primary pickup position, the secondary pickup position/production line must also be given a name if enabled. The value for the production line name defaults to ‘Line 2’ for the secondary pickup, please rename it to a name that makes sense to the operators if dual product mode is enabled.

To calibrate this second pickup position, repeat the calibration steps that were completed for the primary pickup using the same calibration box. Use the scroll bar above the ‘Next’- and ‘Back’-button or drag the screen from left to right to move between the calibration cards.

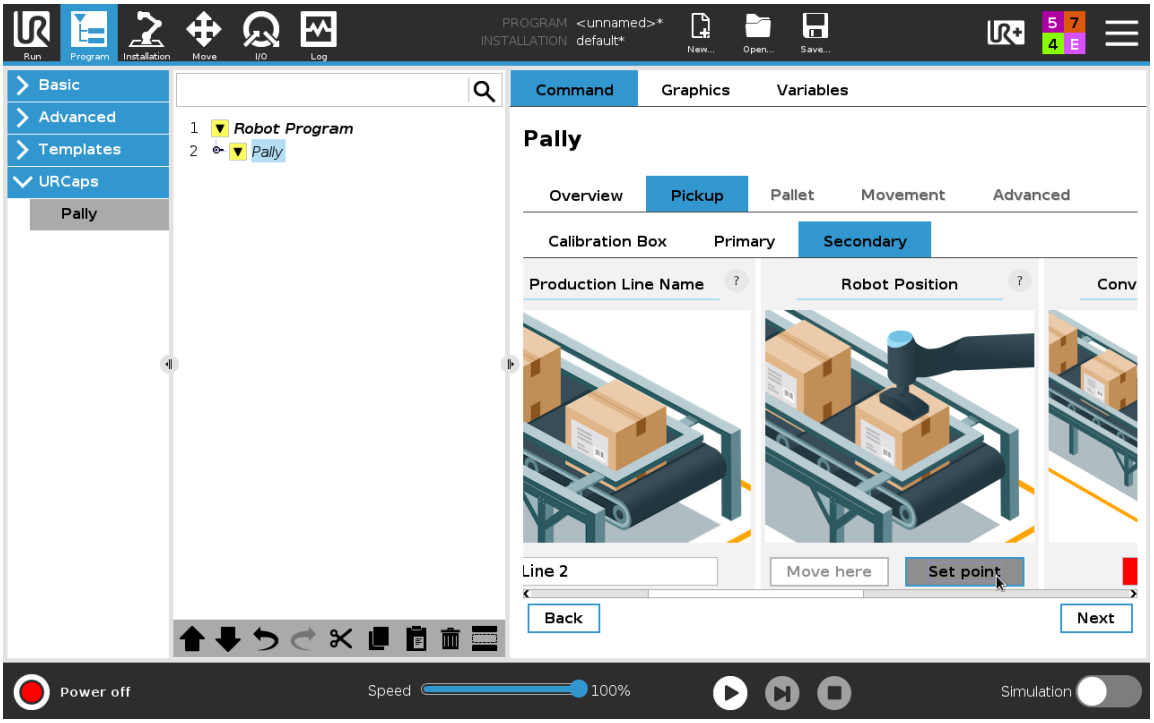


Figure 57: Press the ‘Set point’-button to set the second pickup position

6.2.2 - Pallet

The pallet center point and the floor inclination angle are calibrated by moving the robot above three corners of an empty pallet. These corners are shown in **figure 25**.

Note: The lifting column has to be at its zero position during calibration.

Note: It is necessary to calibrate the pallet positions again when the gripper dimensions on the *Installation* page are updated.

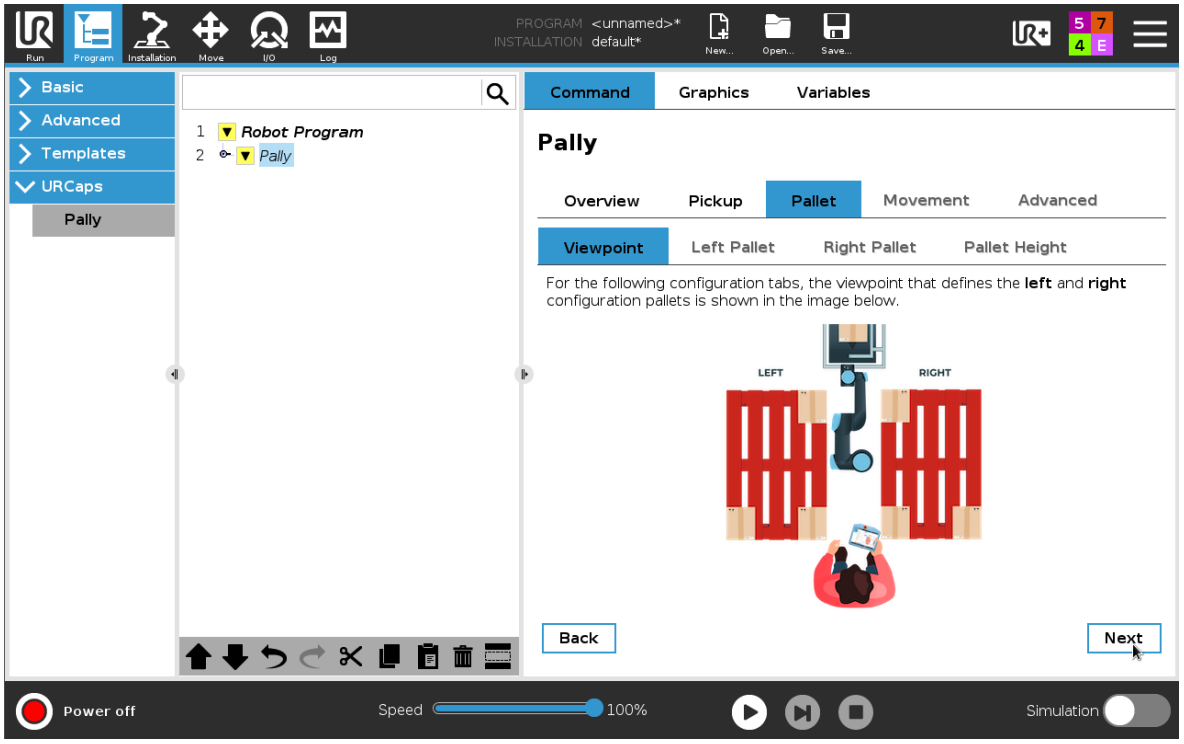


Figure 58: Illustration of the pallet calibration points.

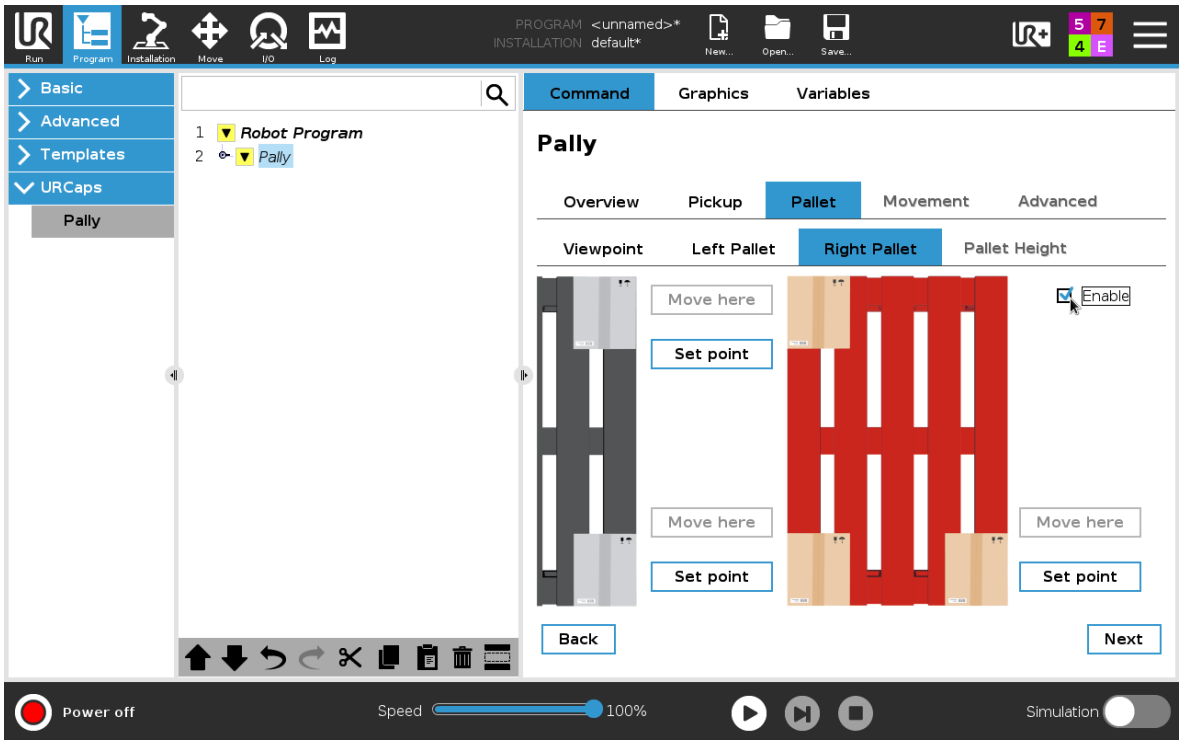


Figure 59: Pallet corners can only be calibrated when the corresponding pallet is enabled.

Measure and enter the height of the empty pallet that was used during calibration, if you want to palletize on pallets with different heights. The floor level will be recalculated according to the values entered here.

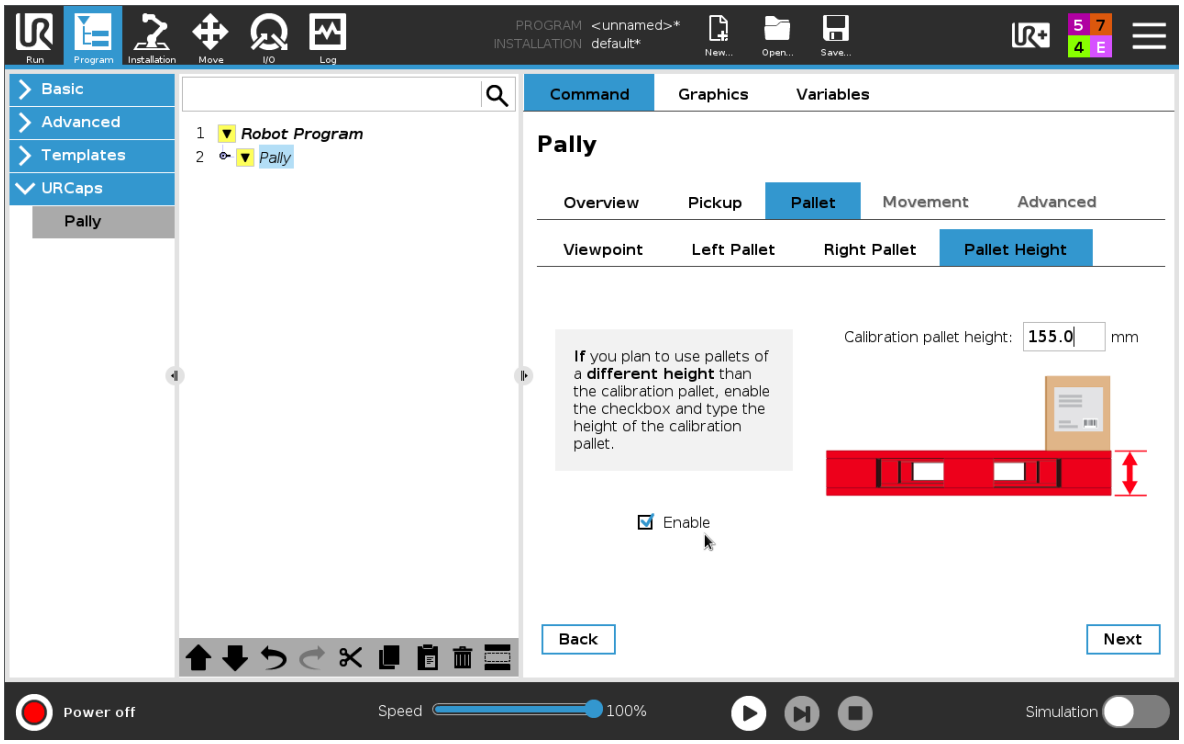


Figure 60: Height of the pallet used during calibration.

6.2.3 - Movement

The speed and acceleration parameters of the palletizing process can be configured on this screen.

Maximum speed and maximum acceleration
These parameters are used to limit the robot movement in general.

Approach speed and approach acceleration
These parameters are used to limit the robot movement when approaching the pallet position. This is especially important when the boxes on the pallet are very close to each other or the box dimensions have large variations, and quick movements would increase the occurrence of unexpected safety stops.

Smart acceleration
Use this setting to control how much the robot can limit the Maximum Acceleration value when moving large and heavy boxes. Too high acceleration may result in frequent safety stops, inaccurate positioning, or dropped boxes. The following options are available:

- Fastest: the robot uses Maximum Acceleration whenever possible
- Fast: acceleration is limited for very large and heavy boxes on a very small gripper

- Medium: acceleration is limited for large and heavy boxes on a small gripper
- Slower: acceleration is limited for large or heavy boxes and/or a small gripper
- Slowest: the most restrictive setting that reduces the acceleration significantly.

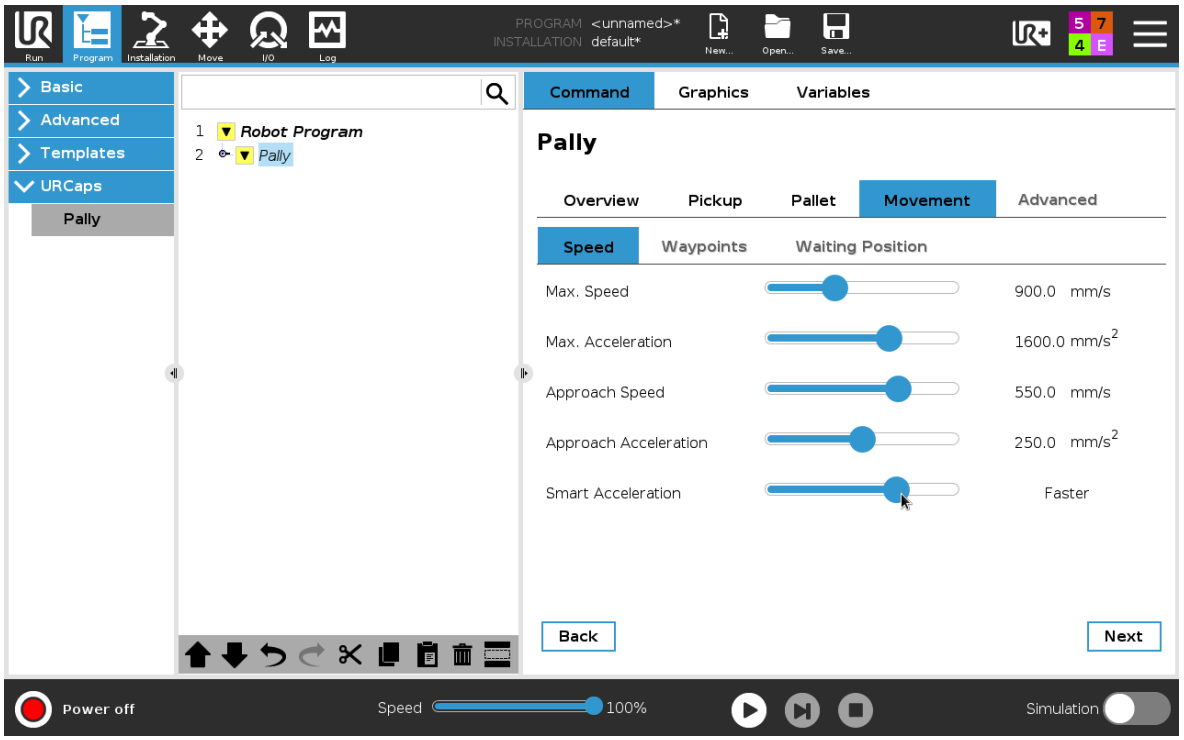


Figure 61: The speed and acceleration of the robot can be tuned to fit the environment.

Note: Always use the lowest possible values that are suitable for the specific palletizing task. High speeds and accelerations may shorten the robot's lifespan and introduce additional safety hazards.

Note: To avoid serious injury, always use low values if the robot handles products with sharp edges! A risk analysis must always be performed based on the products the robot is handling.

Box Free Height

The elevation required for moving the box above the side guides and sensors (**figure 62**). Select 'fixed' to lift each box type to a constant height regardless of the box dimensions (e.g. when lifting over sensors and side guides only). Unselect 'fixed' to lift boxes above other boxes on the conveyor.

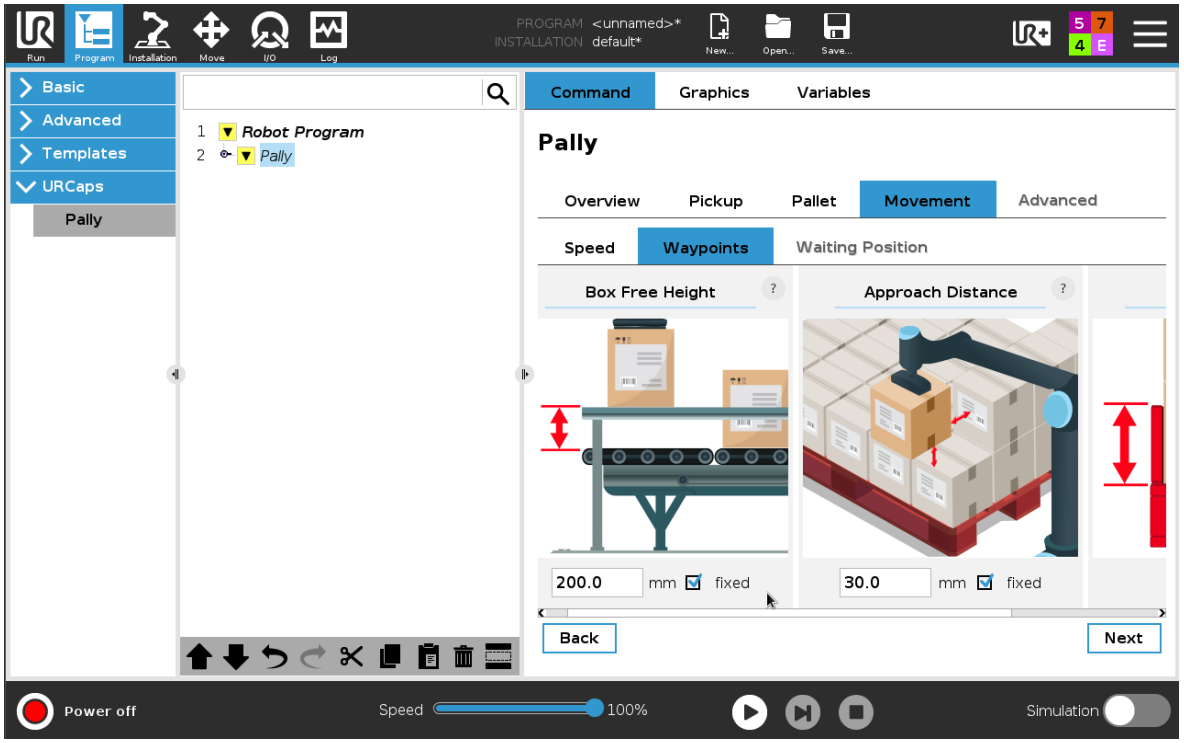


Figure 62: Box free height, the first section of the robot movement.

Approach Distance

Defines the length of the last diagonal movement that is used to put the box on its final destination on the pallet (**figure 63**) with low speed and acceleration. The actual approach distances may be recalculated by the program in run-time when rotation between the pick position and the target position is required. By selecting 'fixed' the program is enforced to finish all box rotations earlier, and approach the target position to the exact distance.

Note: The 'fixed' approach option may have a significant impact on the path planning.

Pallet Lip

Enforces a vertical approach with the specified length on the first layer of boxes (**figure 63**).

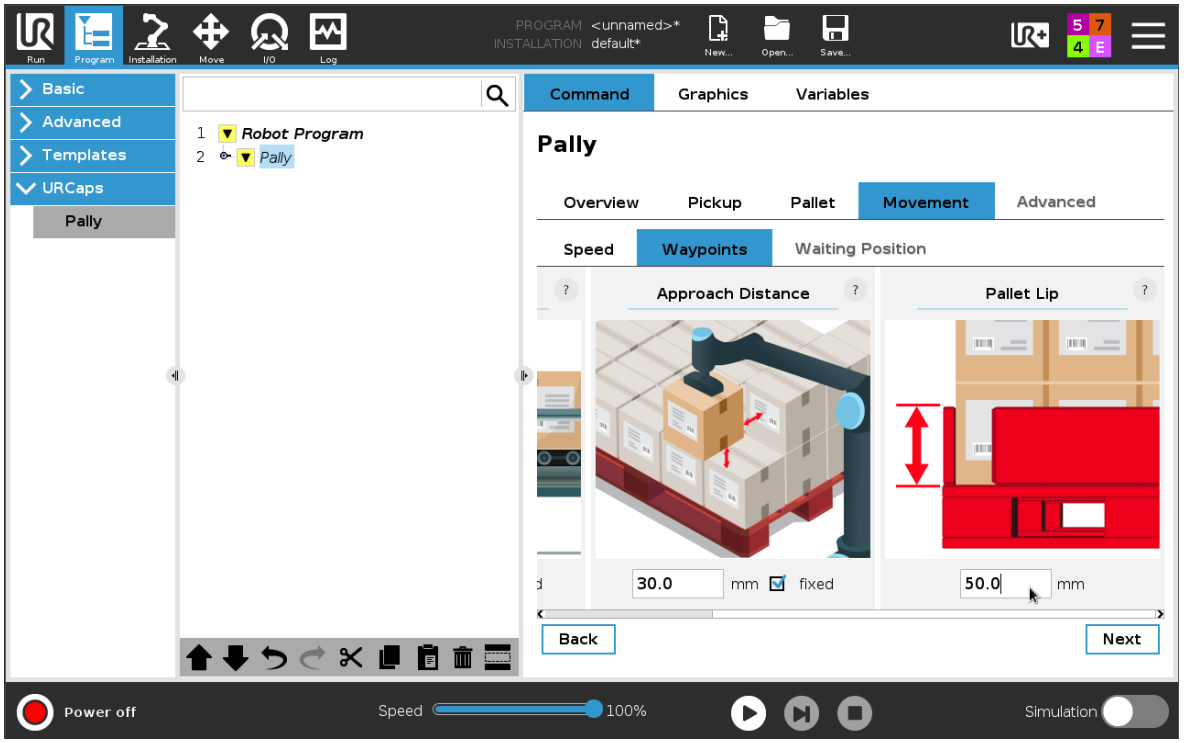


Figure 63: Approach and pallet lip are the last robot movements with the box.

Waiting Position

This is the idle position for the robot while it is waiting for an empty pallet or a product. It is recommended to define this point further away from the pickup position, leaving some space for the operator to reach the conveyor when manual intervention is needed.

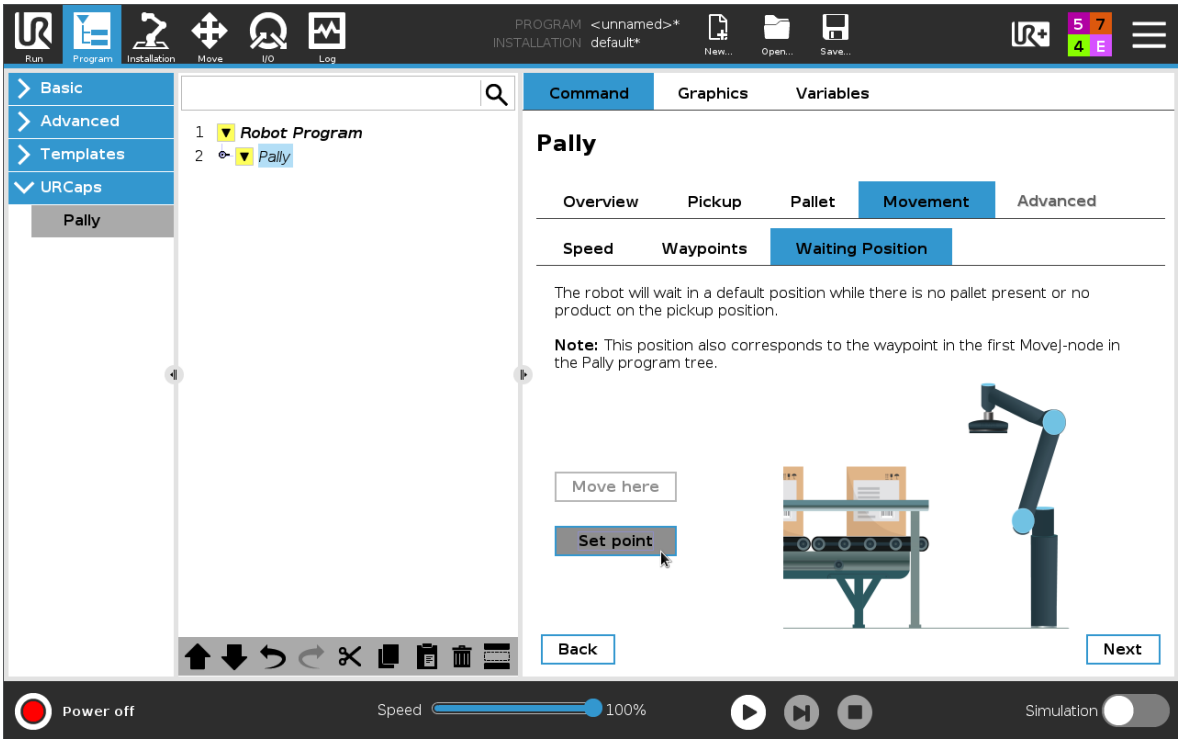


Figure 64: Default waiting position when the robot is idle.

6.2.4 - Advanced

This view contains advanced tuning parameters for the Pally URCap. The view is further divided into "Path planning", "Verified patterns", "Language" and "System" pages.

6.2.4.1 - Path planning

The Path planning view is for tuning the path planning algorithms of the Pally URCap.

Note: Modifying these parameters will fundamentally change the program behavior.

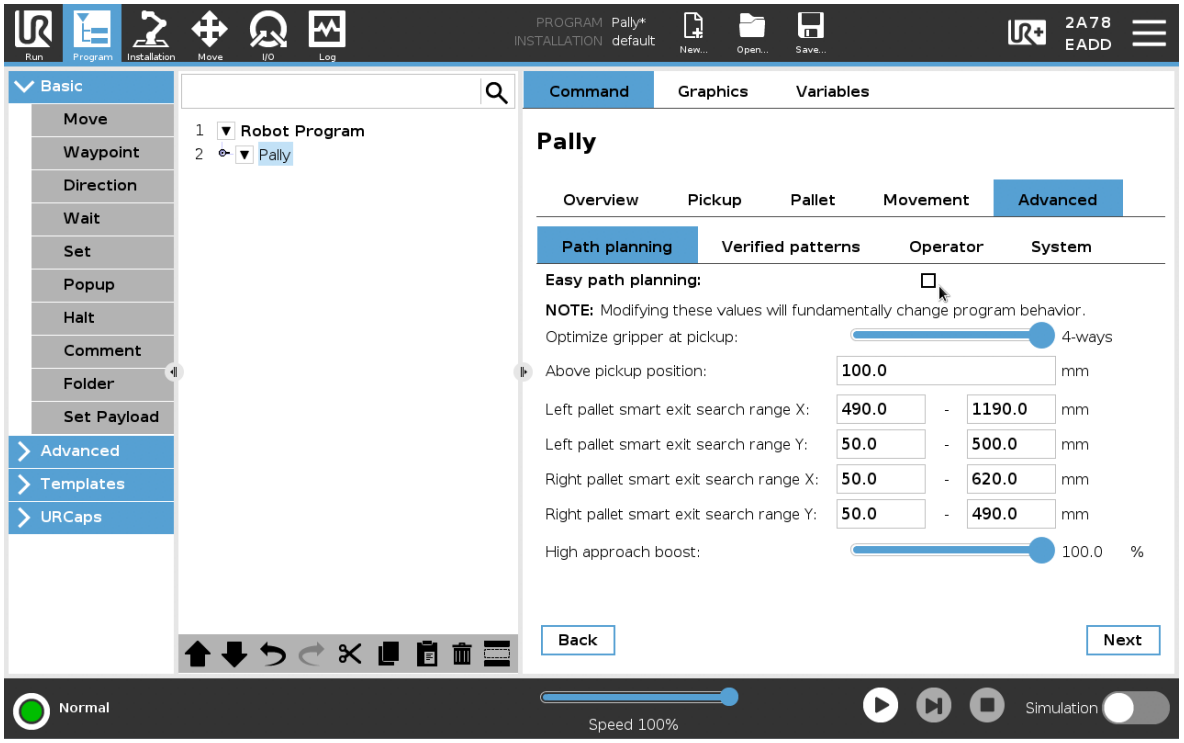


Figure 65: Path planning settings.

Easy path planning

Select this option to let the program compute default values for the path planning parameters based on the current calibration points.
When "Easy path planning" is selected, pressing the **Get path planning suggestions** button will reveal the current default settings and make them editable for manual adjustments.

Optimize gripper at pickup

Depending on this setting, Pally will evaluate several different gripper orientations at the pickup position and choose the one that fits best to a specific target position.
This parameter has special importance when the gripper is offset-mounted. In this case the gripper orientation may have a huge impact on reach and performance.

Above pickup

Gripper distance from the top of the box, from where the robot will move down vertically until the gripper foam reaches the box surface. At this point the program automatically activates compressed air (if available) to clean the gripper foam and the box surface before picking.

Smart exit search range

Defines a rectangular area where the path planning algorithm can search for an intermediate waypoint between box-free and approach position, to perform linear movements without collision to the robot base, see the green areas on **figure 9**.

When entering the smart exit search values, consider the following coordinate system: X=0, Y=0 is identical to, or straight above the *box-free* position. Positive Y will move the box along the conveyor direction, regardless of the current conveyor and gripper orientation. Positive X will move perpendicular to the conveyor towards the *actual* pallet.

Note: The path planning algorithm will exclude smart-exit waypoints that collide with the robot shoulder, and - when moving below the conveyor level - it will also exclude waypoints that collide with the conveyor. When multiple waypoints within the given search area are possible, the waypoint with the shortest total path length will be chosen.

High approach boost
This value must be between 0% and 100%. It determines how high above the other boxes the alternative path planning algorithm moves the box. Lower values let the box move closer to the top of the other boxes on the pallet, higher values keep the box closer to the *box free* height (**figure 11**).

6.2.4.2 - Verified patterns

This is an overview of the waypoint database status according to the current path panning and calibration settings. Press the "TEST" button to refresh the list at any time (this may take seconds to minutes depending on the number of patterns)
The database status can be one of the following:

- Offline path storage is not available or not valid for this pattern.
- Partly tested with other calibration parameters.
  Cached waypoints may require further testing with another URP program.
- Fully tested with other calibration parameters.
  Cached waypoints can't be used with this URP program, but may be used with another one.
- Partly tested with this calibration. (Test in progress?)
  Cached waypoints will be used where available, but further testing is needed.
- Fully tested with other path planning parameters.
  Cached waypoints will be used by this URP program for all box positions.
- Fully tested with the current path planning parameters.
  Cached waypoints will be used by this URP program for all box positions.

Press the "Show details" button to get more information about the pattern status.

Press the "Reset selected" button to clear the waypoint database for the selected pattern and allow the palletizer program to store waypoints again next time the pattern is being palletized.

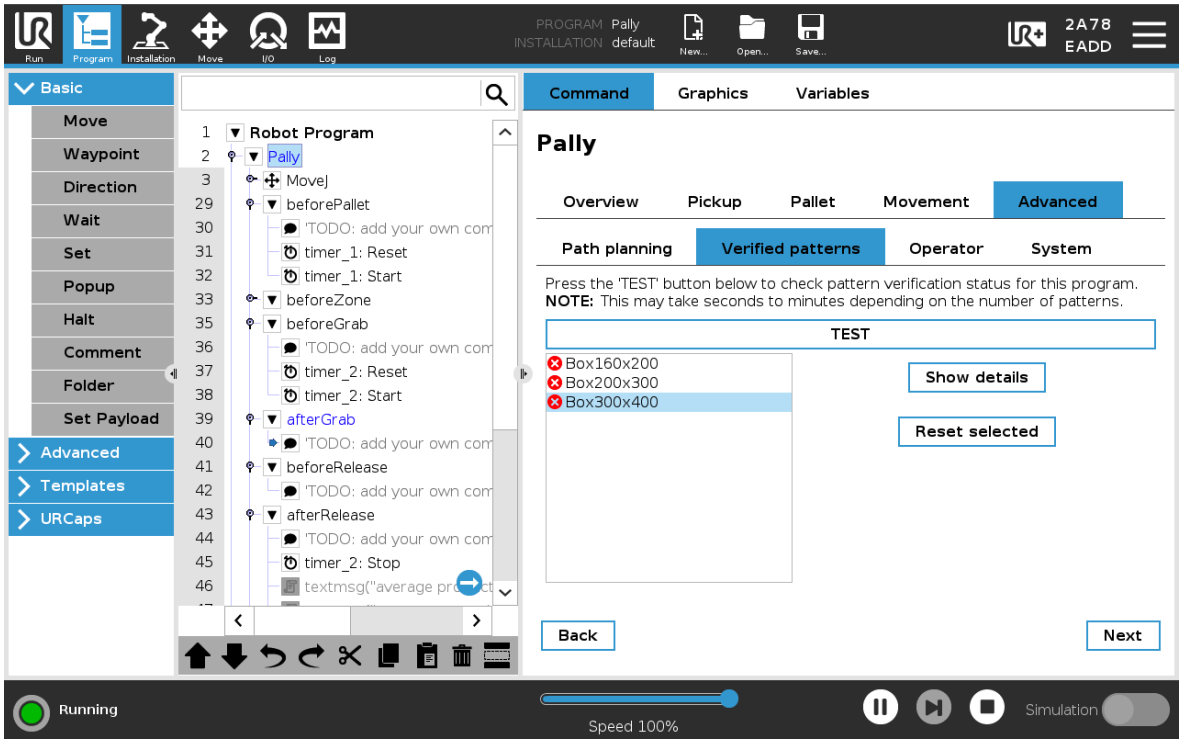


Figure 66: Pattern verification status according to the current program settings.

6.2.4.3 - Operator

Operator interface

Choose the graphical interface that will be presented for the operators. The following options are available:

- None: the program will not present an operator interface. Pattern selection and program flow must be controlled by special variables.
- Standard: 'product selection' and 'continue existing pallet' dialogs are available at program start. No progress indicator is available during the program run.
- Complete: product selection, continue existing pallet, and runtime progress indicator (POP) with pallet confirmation buttons are available.

Operator language

Select the operator language for the Product Selection, Continue Existing Pallet, and optionally the Pally Operator Panel (POP). Currently the following languages are available:

- Czech
- Dutch
- English

- French
- German
- Hungarian
- Italian
- Norwegian
- Polish
- Portuguese
- Slovak
- Spanish
- Swedish

Changing the operator language will not have any effect on the Installation and Program pages. Runtime log messages will not be translated.

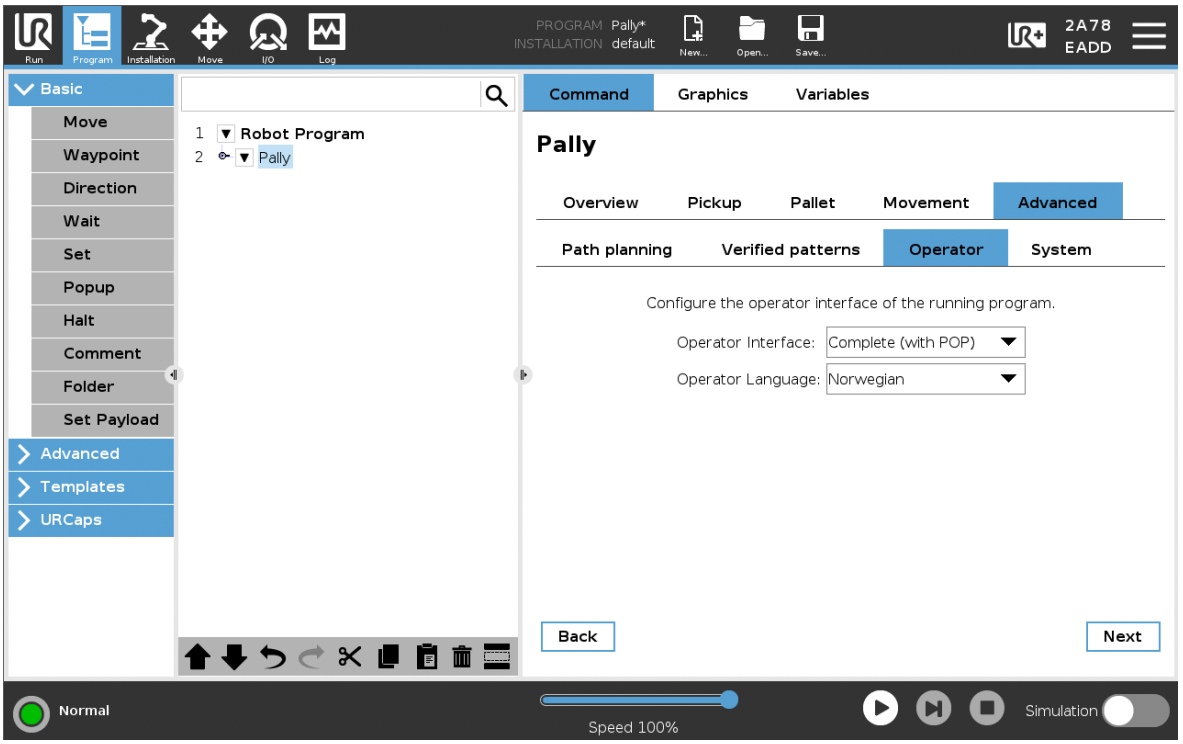


Figure 67: Pally operator interface settings.

Tool button location (CB-series only)
Polyscope version 3.x does not support the UR+ toolbar on CB-series robots. Choose the location of the *functionally equivalent* 'Pally' button (top left, top right, center) from this list.

6.2.4.4 - System

Settings on the System page should not be modified until it is explicitly requested by the support team to do so.

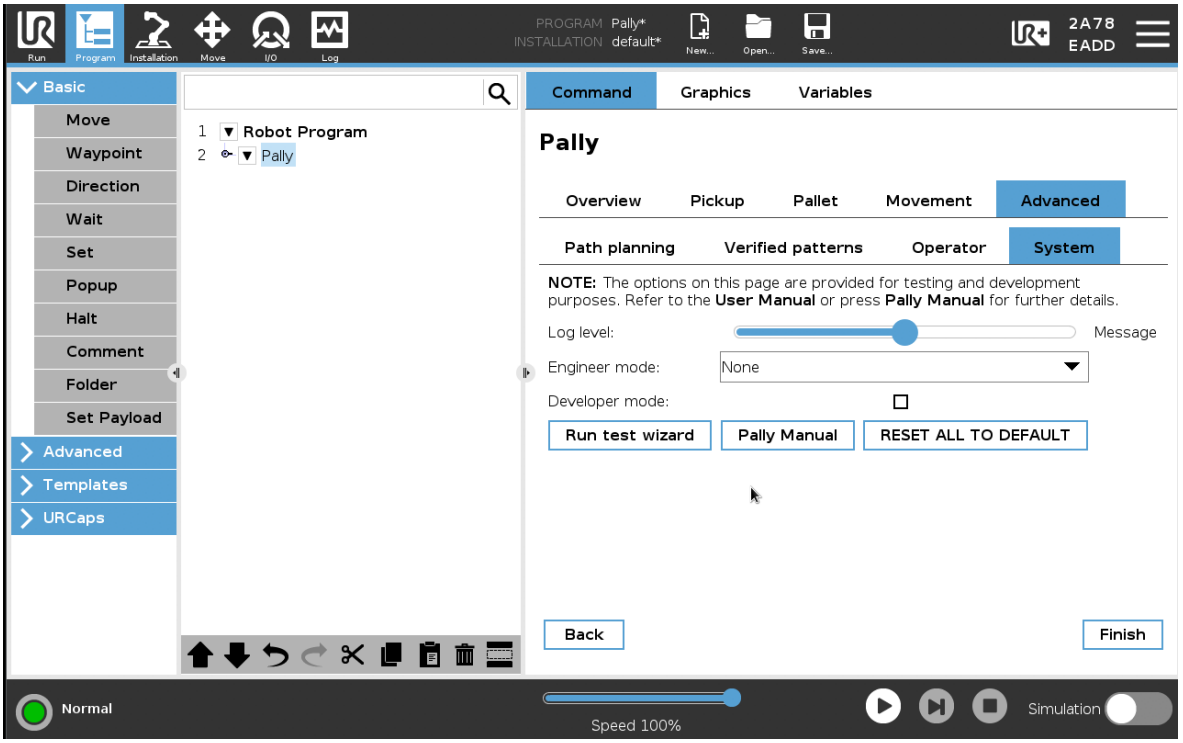


Figure 68: Pally system settings.

Log level

This parameter defines how much information is written into the log file about path planning and robot movement.

- Error: only errors will be written in the log. Usually the program stops on error.
- Warning: errors and warnings will be written in the log.
- Message: errors, warnings, and short summary of the palletizing steps will be written in the log.
- Info: errors, warnings, summary of the palletizing steps and additional details about each calculation and robot movement is written in the log.
- Debug: everything is written in the log.

The default value is "Warning", which means that only errors and warnings are reported.

Note: Log level 'debug' is only valid for the current session and will be limited to 'info' when the program is reloaded or the robot is restarted.

Engineer mode

Reserved for development and support. This option may be hidden in typical setups.

Developer mode

Reserved for development and support. This option may be hidden in typical setups.

Run test wizard
Generates and starts a test script that is used to verify the calibration data. Read more about the Calibration test wizard here.

Pally Manual
This user manual will appear on the teach pendant.

RESET ALL TO DEFAULTS
All settings in the Pally program will be set to their initial value.

6.3 - Callbacks

If additional functionality is required, programs may be added under the callback nodes. E. g. to control warning lights, synchronize other machines by digital outputs, or even send parameters via network interface. Each callback node is invoked at different steps of the palletizing sequence as listed below.

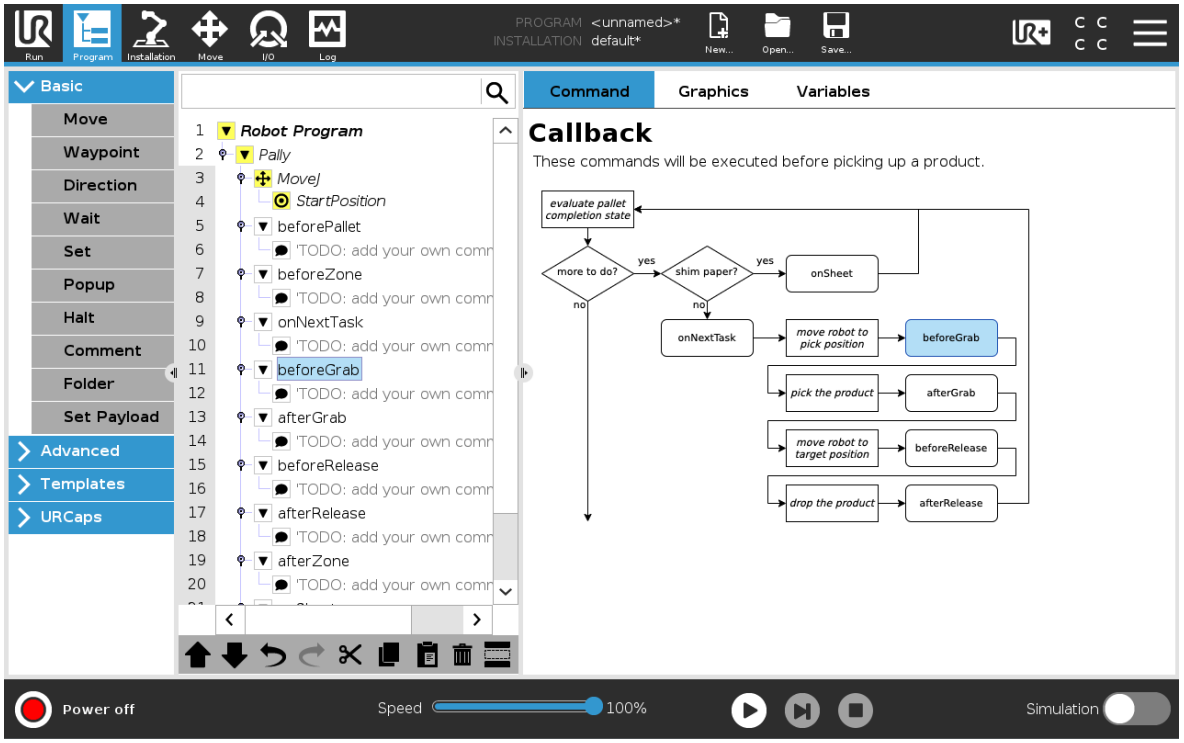


Figure 69: Each callback entry point is illustrated on a flow-chart.

beforePallet: Executed every time before a new pallet is started. This node is typically used to report the status to a production system.

beforeZone: Executed before entering a new zone. Zones are mainly used with lifting columns.

onNextTask: Executed before calculations for the next box position begin. The calculation parameters can be altered here. Set MovePerformed=True here in order to use a custom path towards the pick position.

beforeGrab: Executed when the robot is ready for pick up at the pickup position. Typical uses for this node is to turn on a custom gripper or stop the conveyor. Commands that move the robot to the pick position should be inserted here when a custom path is in use.

afterGrab: Executed when the robot has lifted up the box from the pickup position. Commands that move the robot from the pick position to the target position should be inserted here when a custom path is in use.

beforeRelease: Executed when the robot is ready to drop the box on the pallet. Use this node to turn off a custom gripper.

afterRelease: Executed when the robot has released the box. Use this node e.g. to report the progress to a production supervisor system. Commands that move the robot back from the target position should be inserted here when a custom path is in use.

afterZone: Executed after leaving a completed. Zones are mainly used with lifting columns.

onSheet: Executed every time when a shim paper needs to be inserted. Commands that pick the shim paper and move it to the corresponding pallet should be inserted here.

afterPallet: Executed when the robot has finished the pallet. Use this node e.g. to turn on a warning light, or show a popup message, so the operators can replace the pallet.

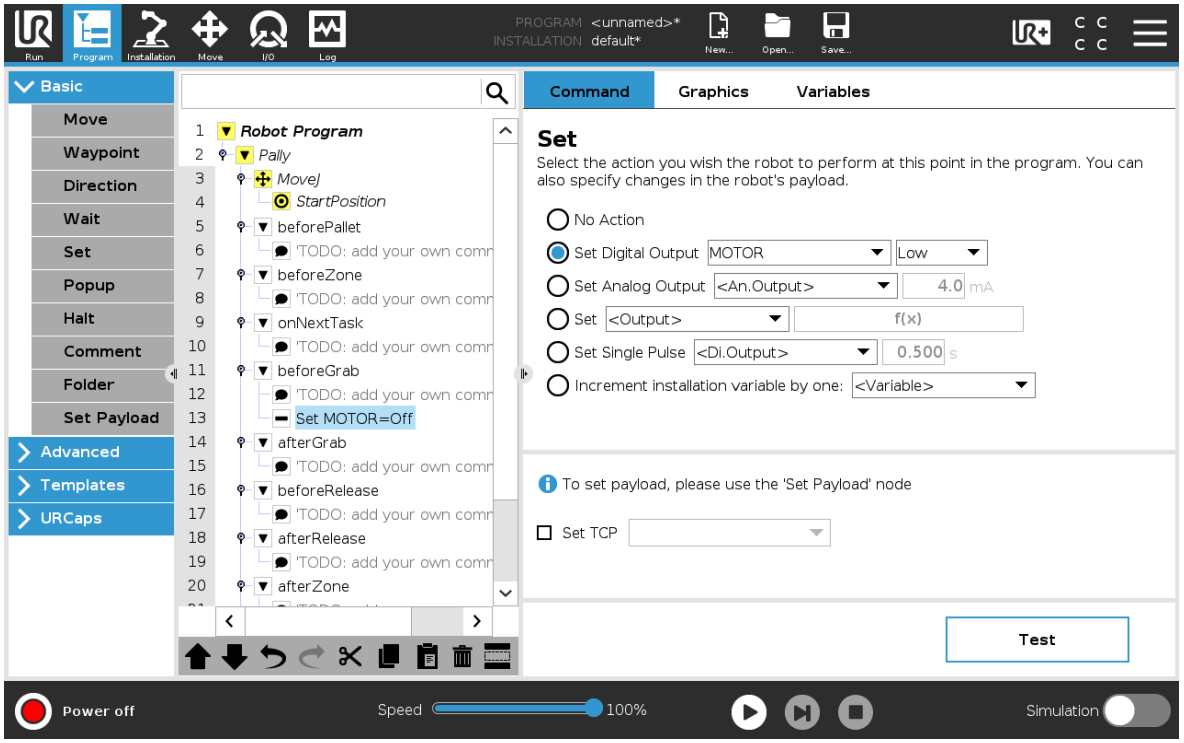


Figure 70: Example of a custom command, before picking up a product.

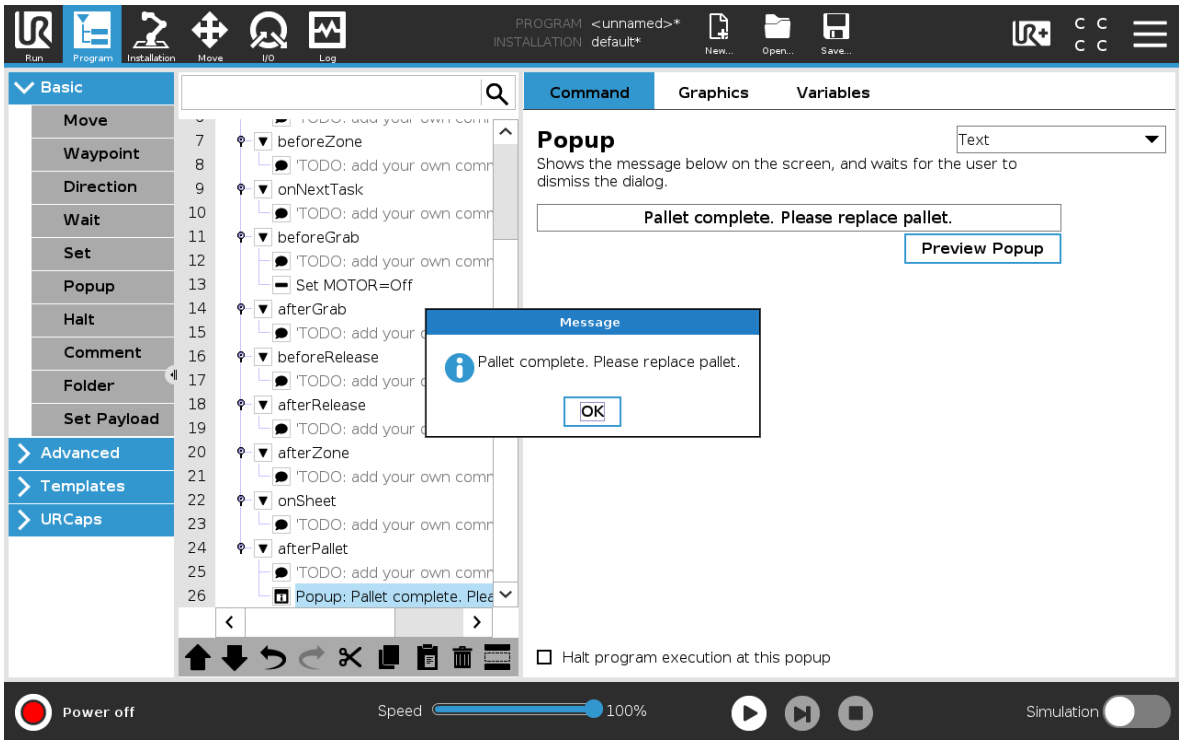


Figure 71 : Example of custom message after a pallet is completed.

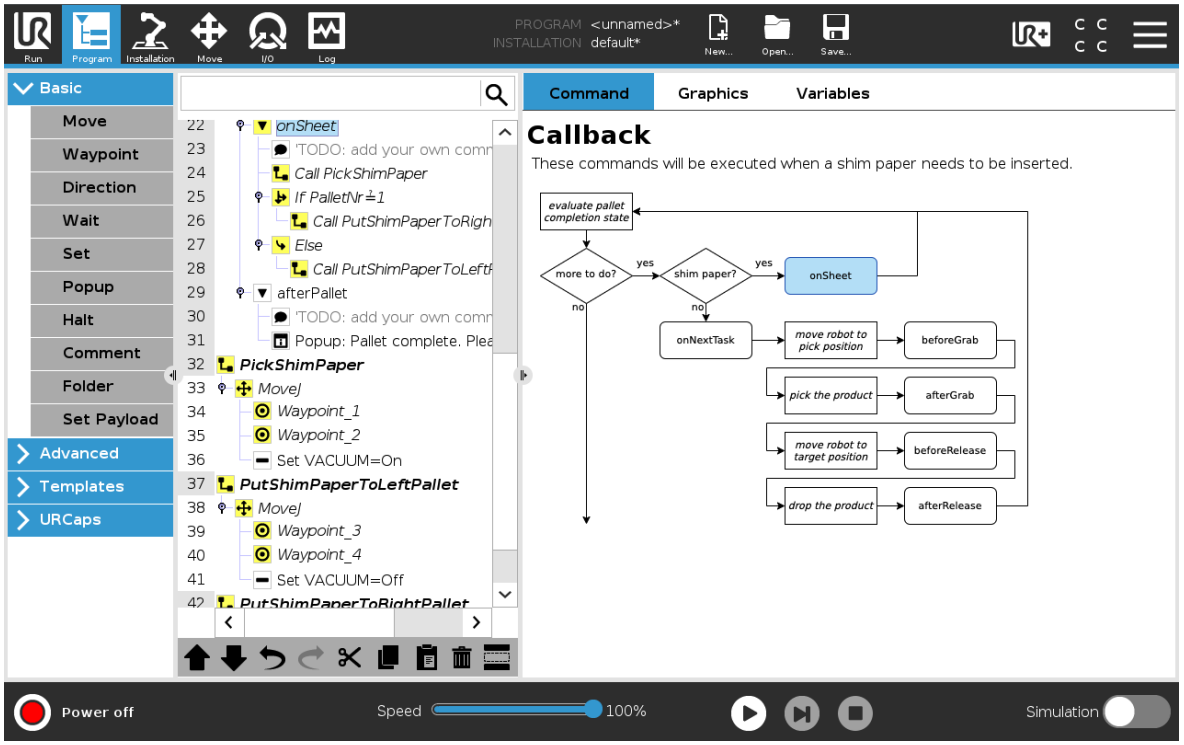


Figure 72: Shim paper functions can be organized into subprograms.

6.4 - Examples: typical configuration scenarios

6.4.1 - Classic setup with one optimal pickup position

The recommended configuration is shown in **figure 73**. There is enough room around the pickup position where the robot can move the product towards the pallets. Usually no additional configuration is required.

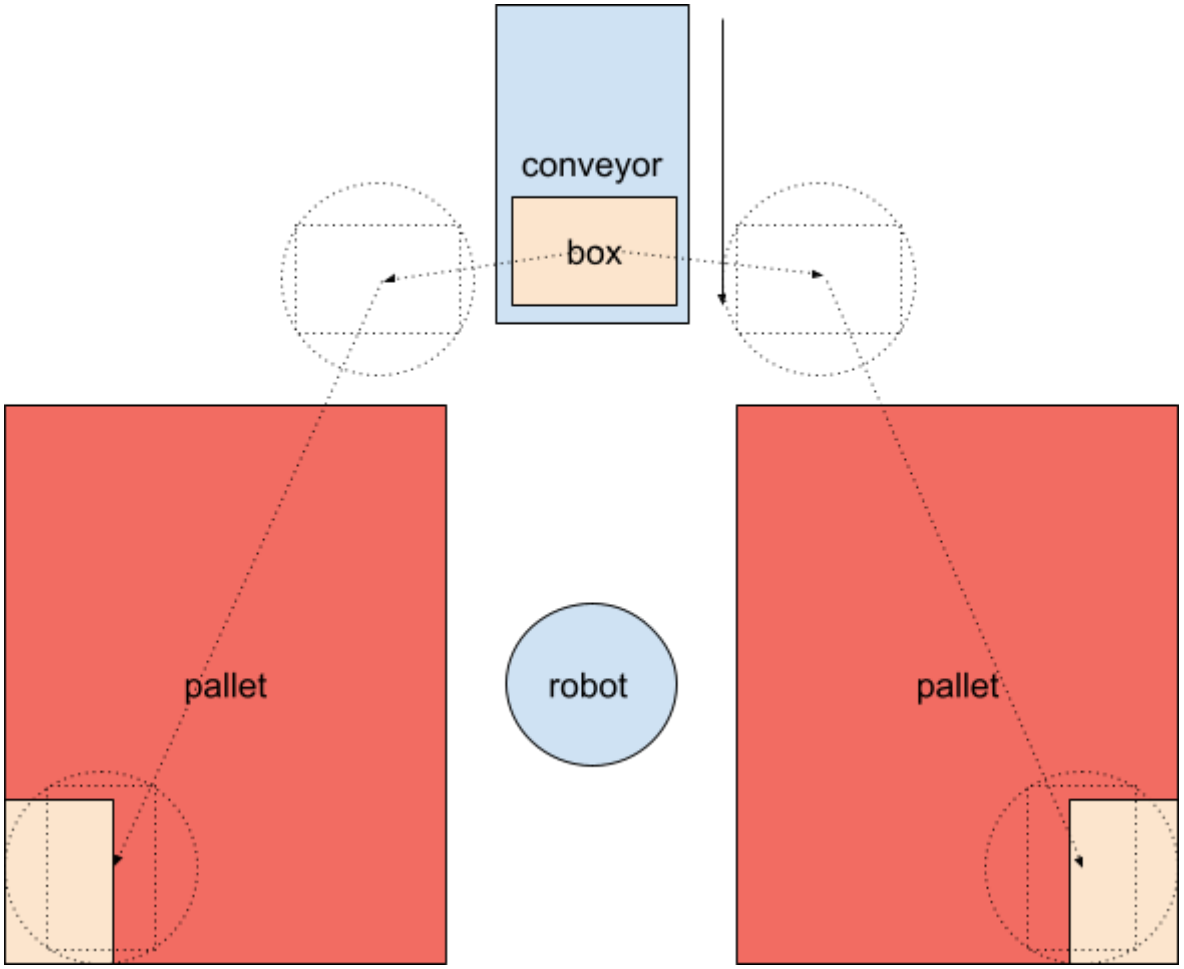


Figure 73: Classic setup with optimal pickup position.

6.4.2 - One pickup position with products coming in from the side

Setups similar to illustration in **figure 74** introduces a few points to be taken into consideration.

- Pickup position should not be in the pallet shadow, i.e. behind a pallet seen from the robot.
- Use limited smart exit search.
- Choose *box_free* high enough to move the box freely above the next boxes, as shown on **figure 75**.

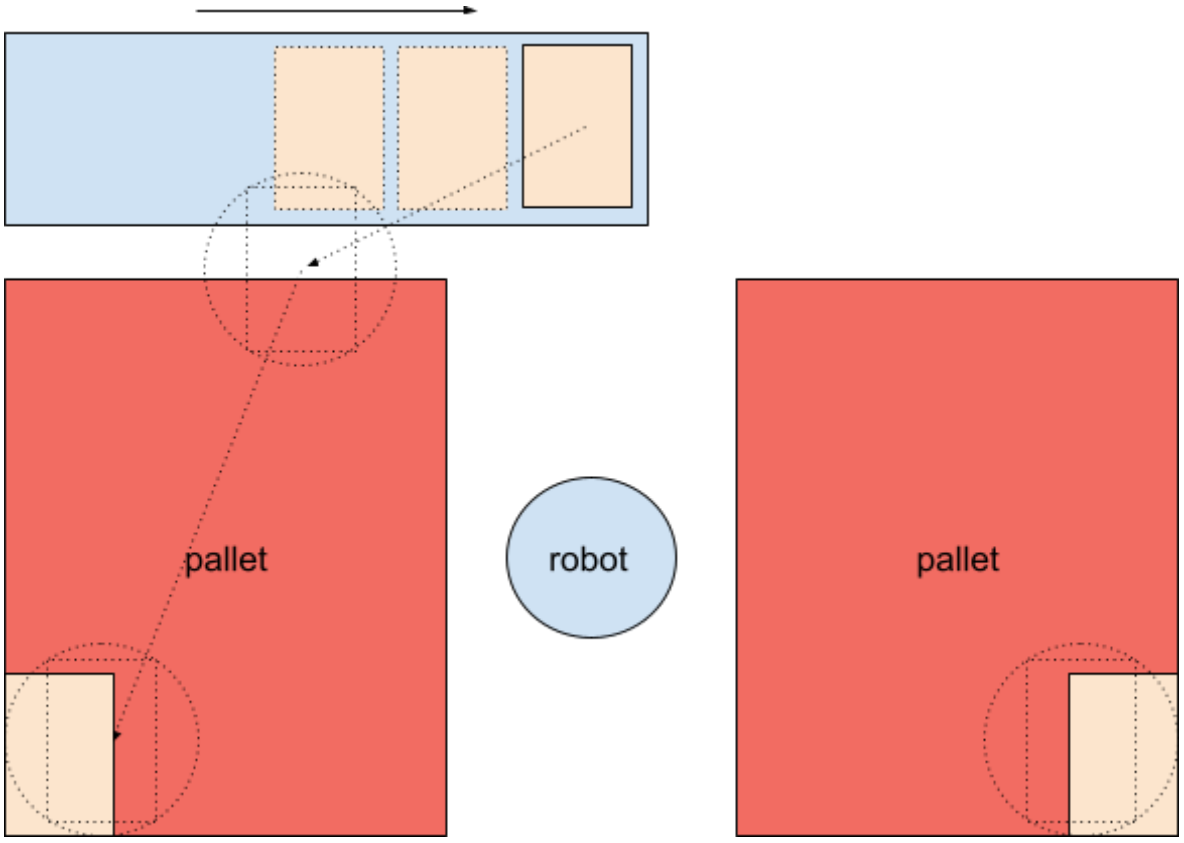


Figure 74: Pickup position sideways. The box will be moved above the other boxes on the conveyor.

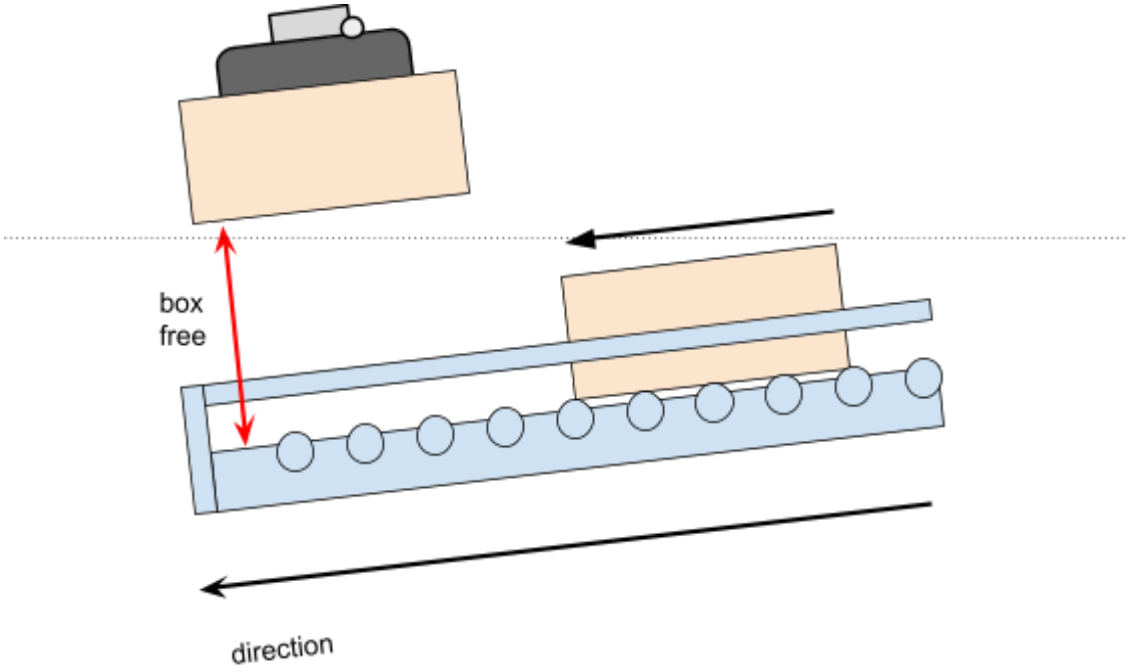


Figure 75: Choose a box free distance that is big enough to move the box above the others.

6.4.3 - One pallet position and one pickup position on the opposite side

Setups similar to illustration in **figure 76** introduces a few points to be taken into consideration.

- Choose *box_free* high enough to move the box freely above the next boxes.
- Use relatively large values for *smart exit min_x* and *negative min_y*.
- Recommended settings (approximately):
    - smart exit search area X: 500 - 1000 mm
    - smart exit search area Y: -900 - -700 mm (negative)
    - box free: 350 mm (or more, depending on the box height)

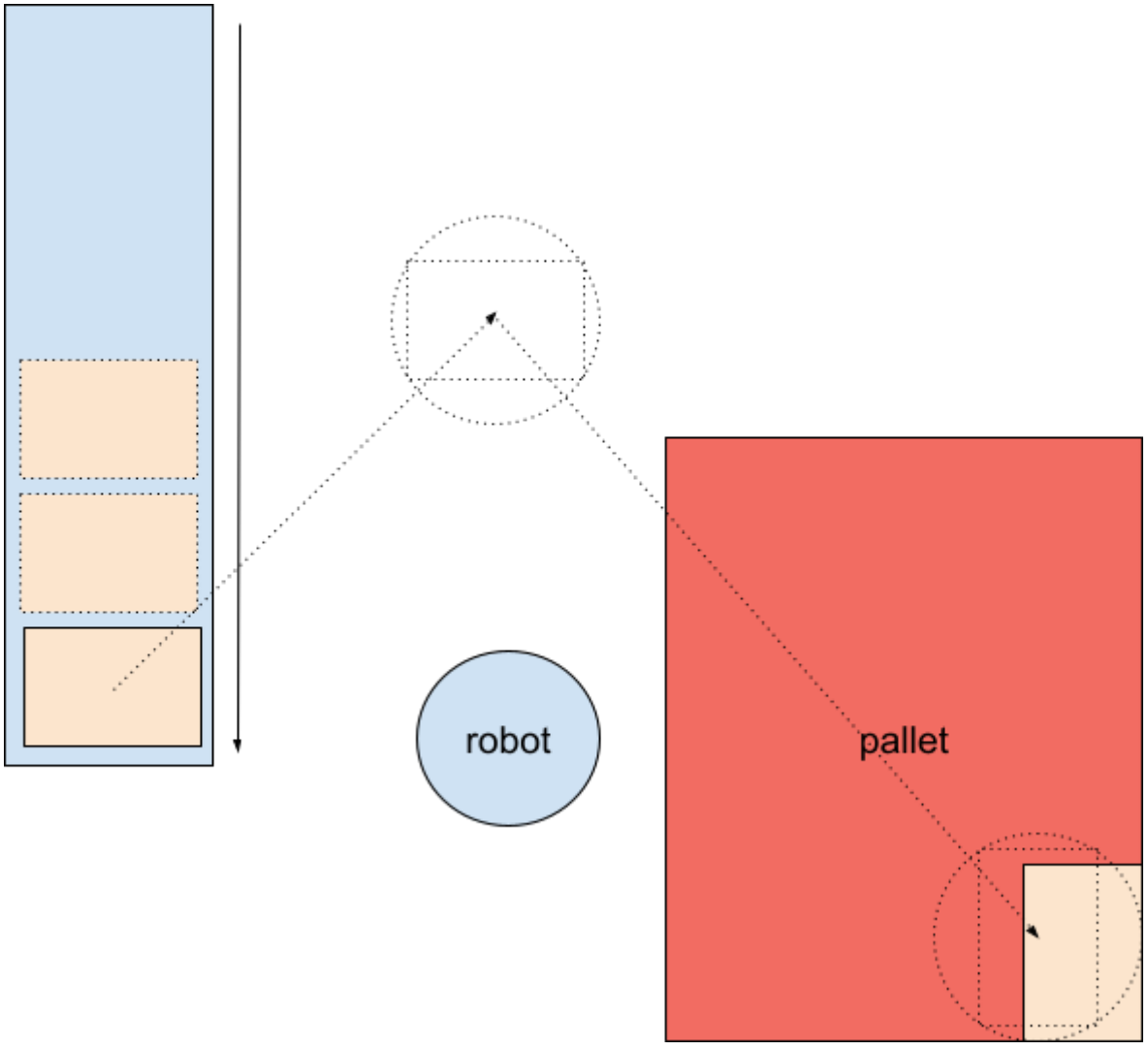


Figure 76: Pickup position and pallet position are on the opposite sides of the robot.

6.4.4 - Two parallel pickup positions

For setups similar to the illustration in **figure 77**, the path planning algorithm needs to be tweaked as described below.

- Fixed guide width for conveyor 2, measured as in **figure 78**.
- Fixed guide width for conveyor 1, measured as *b* in **figure 78**.
- Measure *conveyor total width* from the left side of the left conveyor to the right side of the right conveyor. See total width for both conveyor 1 and 2, measured as *c* in **figure 78**.

Note: Measure fixed guide width from the guide edge to the outer end of the conveyor area - which may be the other conveyor.

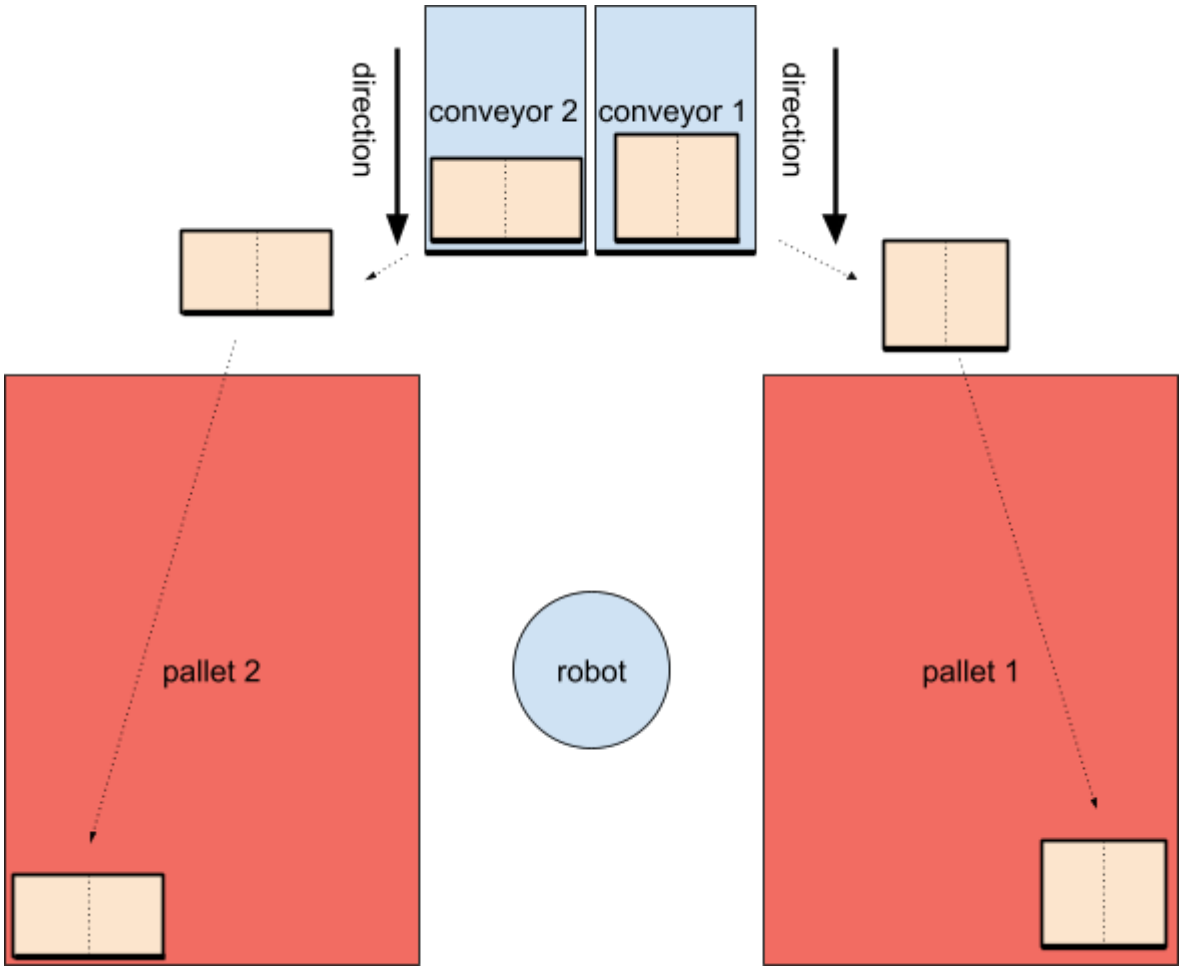


Figure 77: Two parallel pickup positions.

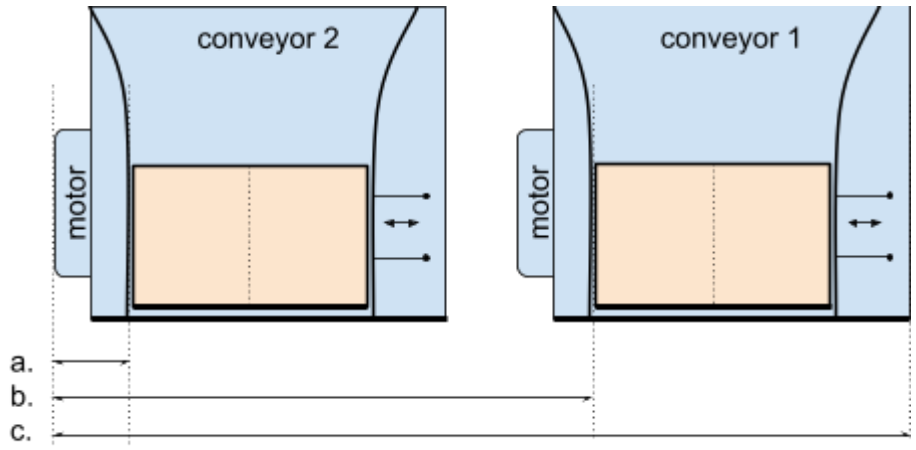


Figure 78: Measure the conveyor dimensions as if they were one big conveyor.

Note: The robot might have to move the boxes above the other boxes on the other conveyor (**figure 79**) when palletizing from one pickup position to the opposite pallet. In such projects, set the *box_free* parameter large enough to move the box above other boxes on the other conveyor if necessary (**figure 80**).

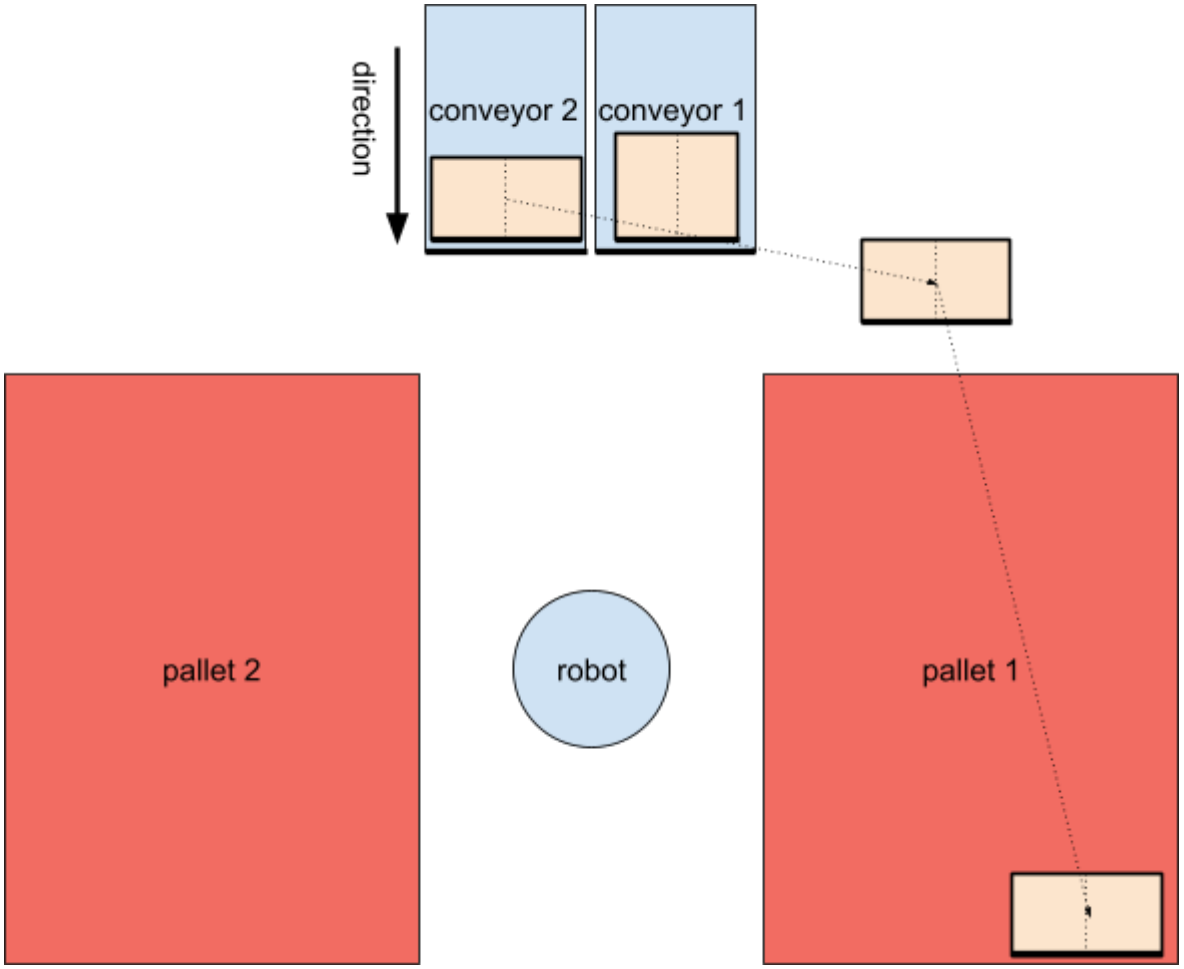


Figure 79: Moving the box above other boxes on the other conveyor.

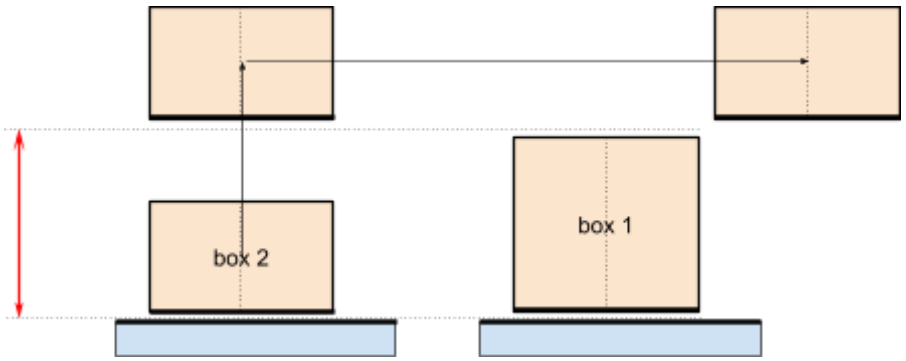


Figure 80: Moving the box above boxes on the other conveyor.

6.4.5 - Two opposite pickup positions

Setups similar to illustration in **figure 81** introduces a few points to be taken into consideration.

- The pickup position should not be in the pallet shadow, i.e. behind a pallet seen from the robot.
- Use limited smart exit search.
- Choose *"box free"* high enough to move the box freely above the next boxes (**figure 75**).

Note: The second box may be in the pallet shadow, making double grip impossible.

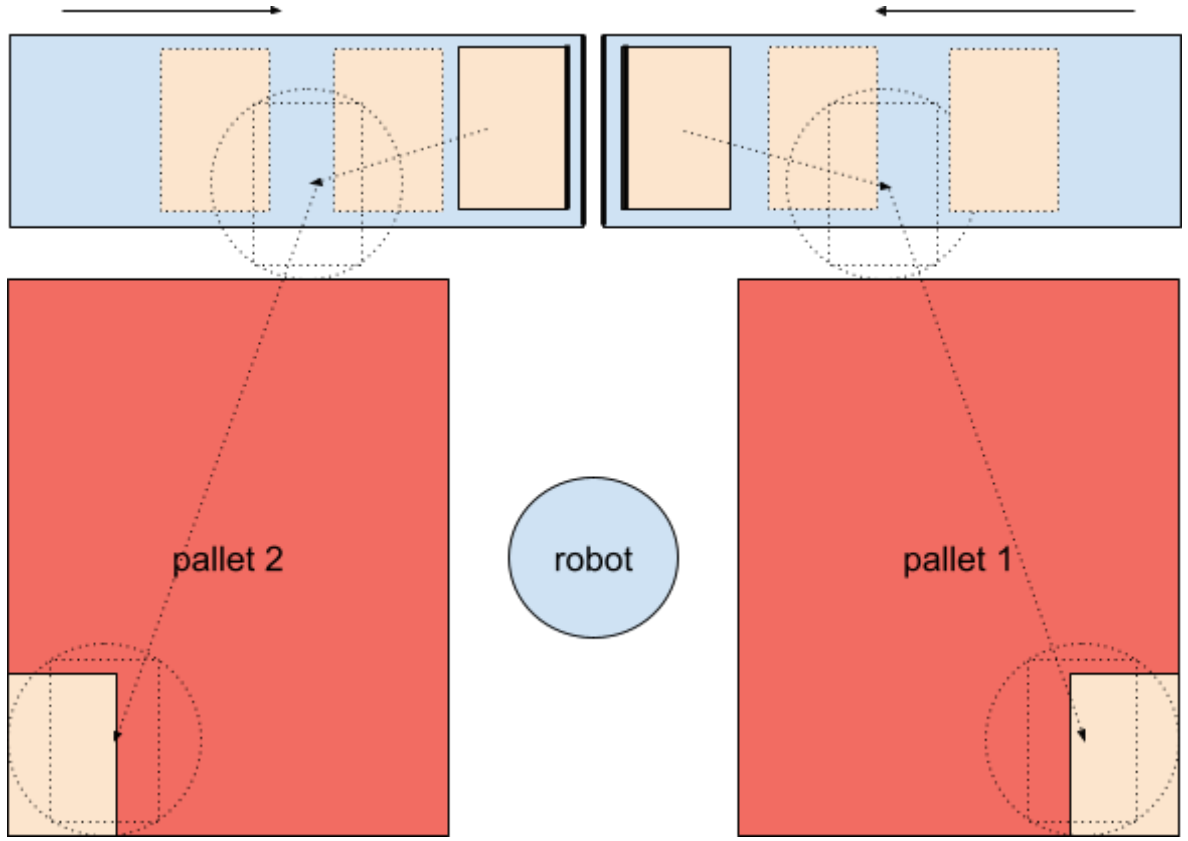


Figure 81: Boxes should travel above other boxes on the conveyor.

6.4.6 - Limited space, walls other object

Setups similar to illustration in **figure 82** introduces a few points to be taken into consideration.

- Individually configure smart exit for left and right pallet.

- The smart exit search area may be limited to one single point or a very small area, to avoid collision with the wall.

Note: Reserve enough distance between the wall and the pallet to perform box rotation.

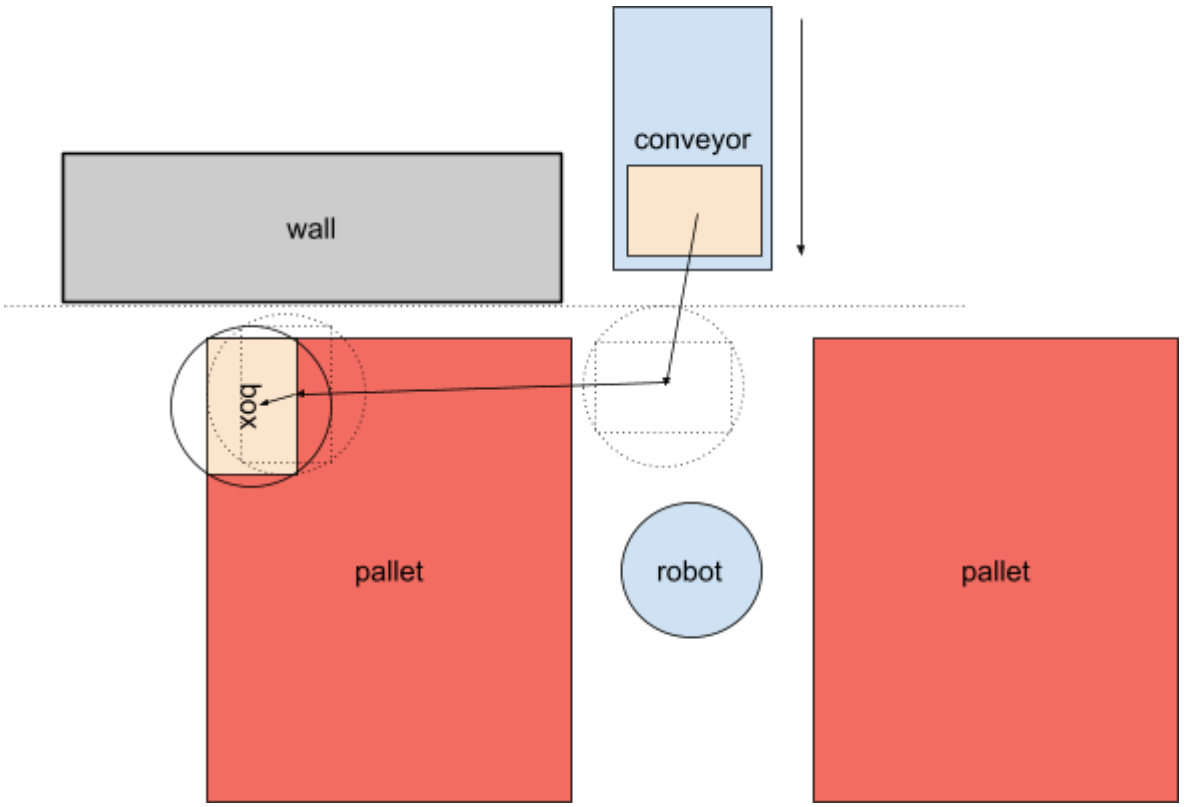


Figure 82: Path planning with fixed *smart exit* position, to avoid collision with the wall.

6.5 - Calibration test wizard

The calibration test wizard is a utility that can be used to verify the calibration points. The primary and secondary conveyor, left and right pallet positions can be tested.

There are several different types of tests included, which are performed sequentially.

- The test wizard performs several calculation routines to identify some of the most typical mistakes usually made during calibration.
- It can move the robot along a path that is generated based on the current calibration data, and ask the user to confirm that the robot motion looks consistent to the physical conveyor and pallet placement.
- The wizard can also utilize force-controlled robot motion to measure calibration accuracy, and make small adjustments based on the measurements.

Before each test step, the wizard explains the test that will be performed next. The user can start or skip the test and proceed to the next step by pressing the corresponding button.

When a test fails, a warning message appears with possible explanations of the failure. The user can stop the test wizard and fix the error, or proceed to the next step.

The test cases are divided into test groups. Currently there are 4 groups:

- Testing the primary pickup position
- Testing the secondary pickup position (in projects where available)
- Testing the right pallet position (in projects where available)
- Testing the left pallet position (in projects where available)

The test cases for the primary and secondary pickup are identical.
The test cases for the left and right pallet are identical.

Testing a pickup position contains the following test cases

- Moving parallel to the conveyor: this test can help detect calibration issues related to TCP settings.
- Moving to the pick position with 180 degrees rotation: this test can help detect calibration issues related to TCP settings.
- Automatic optimization of the pick position: this test can measure the pick position accuracy and adjust the robot position slightly up or down. (see note below)

Testing a pallet position contains the following test cases

- Verification of the pallet size: this test can verify that the calibration box was used as required.
- Verification of the pallet shape: this test can verify that the correct pallet corners were used during calibration.
- Verification of the calibration points: this test can check that the robot position at each pallet corner is accurate enough.
- Automatic optimization of the pallet corner positions: this test can measure the position accuracy and adjust the robot position slightly up or down. (see note below)

Note: When running the tests that use force-controlled motion, the robot will firmly push the calibration box vertically down until a specific force is detected. Make sure your calibration box is made of a material hard enough to tolerate the pressure. Very soft or fragile boxes can lead to inaccurate measurements, measurement failures, or even damage to the box.

---

7 - Using the palletizer program

This chapter will give an introduction on how to use the program. It will describe how to create the main program, how to start the program, how to use the program in different modes, error recovery and shutdown of the robot.

7.1 - Creating the main program

Create the palletizer main program as described in Program settings.
The palletizer program should be saved on the disk as a normal UR program.

Note that there should only be *one* Pally-node in the Robot Program.

7.2 - Starting the program

Start the robot. Click the play button in Polyscope and then select "Run program". A window from Pally will appear, as described in the next section.

7.3 - Using the program

Depending on the installation settings for the ‘dual product mode’ option (this is set to ‘true’ or ‘false’ in the Pally installation node → Advanced-tab → Systems-tab, see 6.1.6 Advanced), one of two things will happen once the robot runs the program. Either the single product mode window will open, or the dual product mode window will open.

Below, the graphical user interfaces for the single and dual product modes are described step-by-step.

7.3.1 - Single Product Mode

In this mode, the robot can palletize 1 product type, from 1 or 2 pickup positions.

The first window that appears once the program is run, is the ‘Pattern Selection’-window (as shown below).

Start a new pallet

- Select product by name
- Press ‘Start Program’-button
- Palletize until program stop

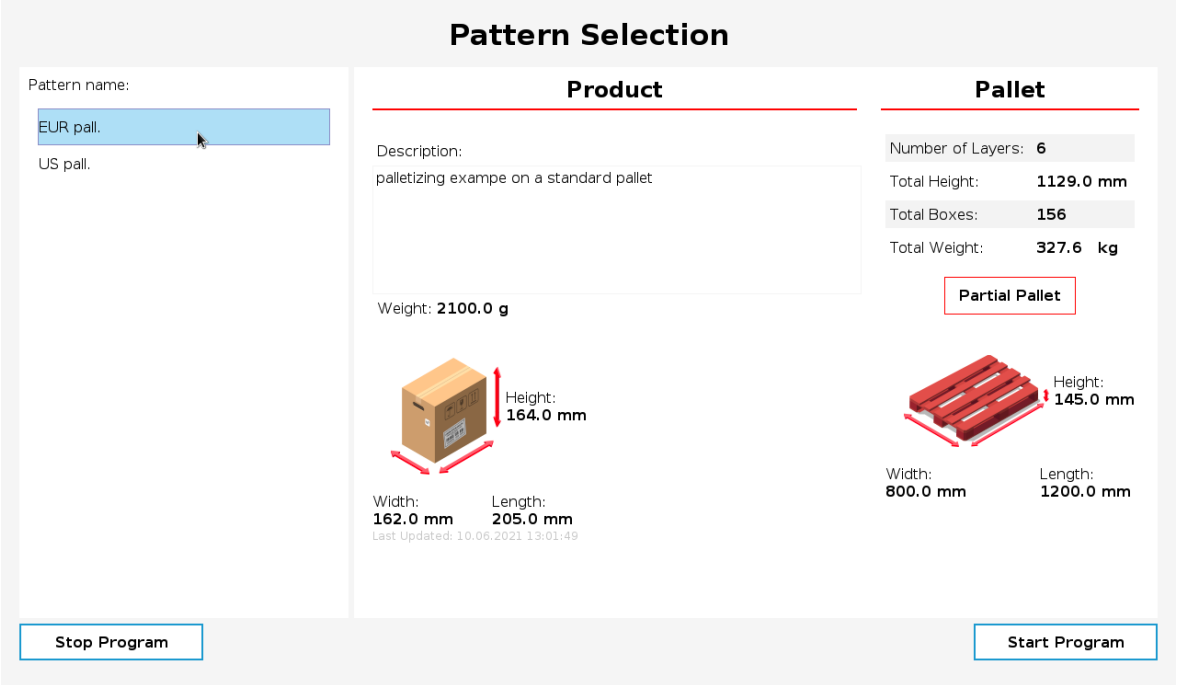


Figure 83: overview of all available patterns on the Pattern Selection dialog.

Select the pattern that is going to be palletized in the ‘Pattern Selection’-window and press the ‘Start Program’-button to start the palletizing process.

Once the robot has palletized *at least* one box on a pallet, and the program is stopped then restarted; then the ‘Continue existing pallet’-window is opened (as shown below). This window shows the progress on the current pallet.

Continue existing pallet

- Press "Continue Palletizing"
- Palletize until program stop

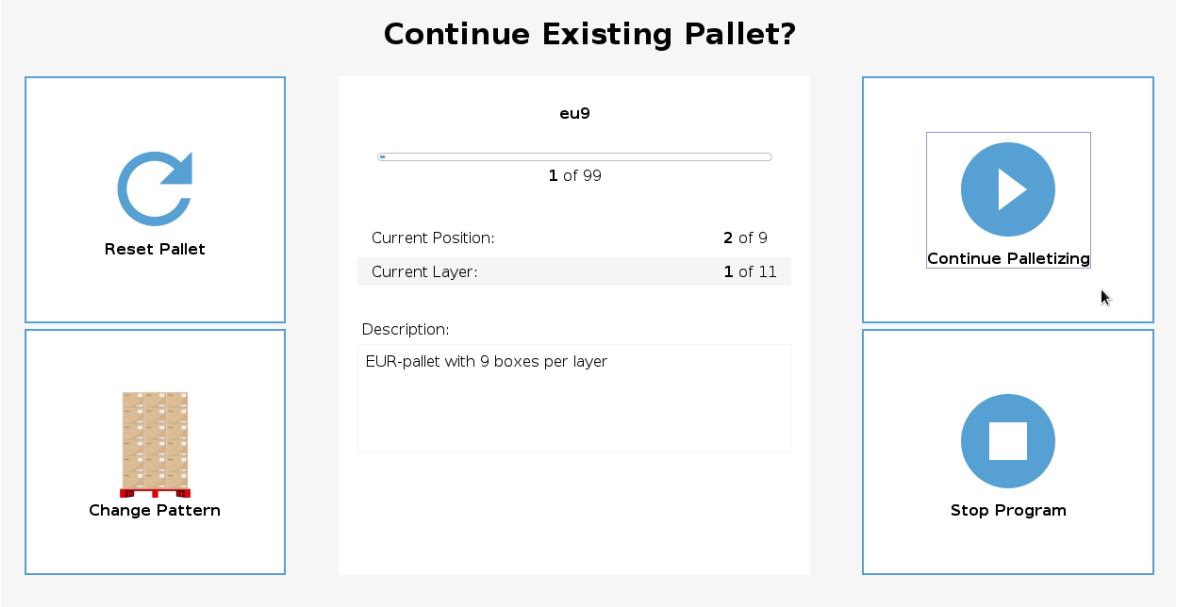


Figure 84: Pallet completion state on the Continue Existing Pallet dialog.

There are four options to choose from in the ‘Continue Existing Pallet’-window. One can either stop the program, or simply continue palletizing the current pallet.

Another option is to reset the current pallet, this effectively means that the palletizing will continue on an empty pallet once the ‘Continue Palletizing’-button is pressed; but it continues using the same pattern as before. Note that either the left or right pallet has to be confirmed before the palletizing will continue after the pallet has been reset.

The final option is to change the current palletizing pattern, this is done by clicking the ‘Change Pattern’-button. Once this button is clicked, the ‘Pattern Selection’-window is opened. Once a pattern is selected in this window and the ‘Start Program’-button is pressed, then the current palletizing process is reset and the newly chosen pattern is run.

7.3.2 - Dual Product Mode

In this mode, the robot can palletize 1 product type from 1 or 2 pickup positions, or 2 different product types from 2 pickup positions, depending on the daily production plan. From hereafter the following names will be used to describe these cases:

1. 1 product type from 1 pickup position = **single *pickup* mode**
2. 1 product type from 2 pickup positions = **dual *pickup* mode**
3. 2 product types from 2 pickup positions = **dual *product* mode**

Please note that the single and dual pickup modes (cases 1 & 2) are possible to achieve *without* having dual product mode enabled (see section 6.1.6 Advanced on how to enable dual product mode). But, to be able to run case 3, i.e two different product types from two

different pickup positions, dual product mode *must* be enabled. All these cases are possible to achieve using dual product mode to allow for variations in the daily production plan.

Examples of various dual product mode setups are described in 6.4.4 - Two parallel pickup positions and 6.4.5 - Two opposite pickup positions.

Note When using a lifting column in dual product mode, the dynamic positioning option must be selected in order to ensure that the lifting column is always in the optimal position.

The first window that appears once the program is run, is the ‘Pattern Selection’-window (as shown below). Unlike the single mode ‘Pattern Selection’-window, the dual product mode ‘Pattern Selection’-window allows the user to choose the pattern that will be palletized form each production line.

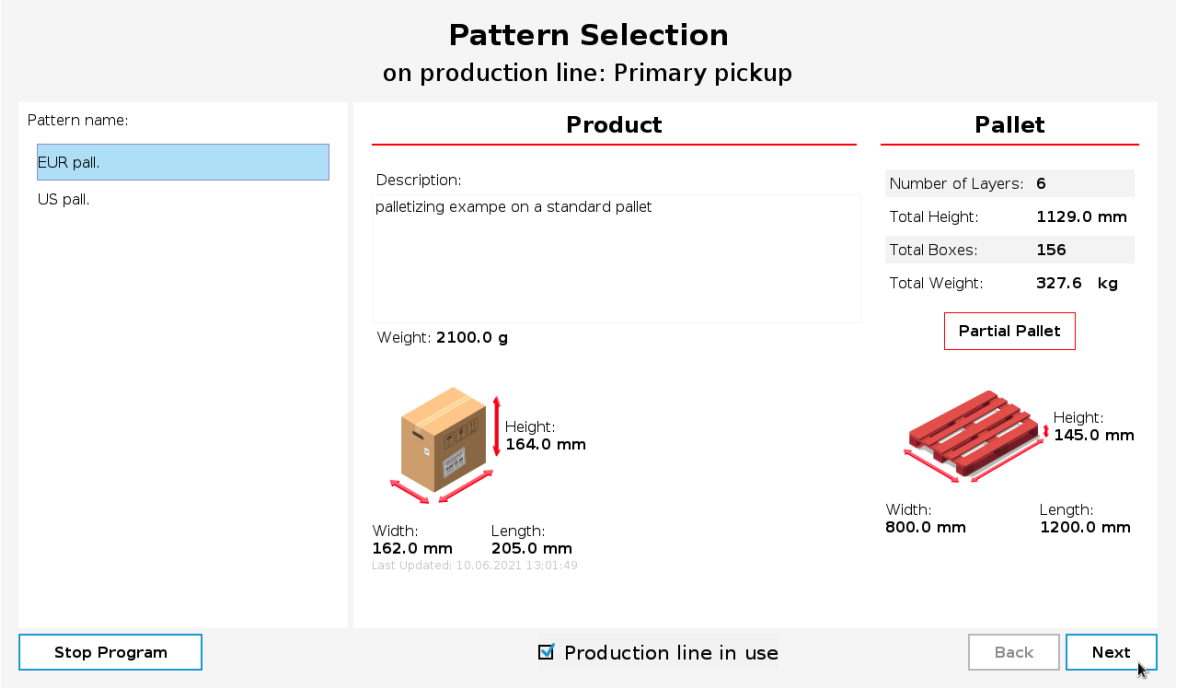


Figure 85: Pattern Selection for the primary pickup/production line in dual product mode

The pattern for the primary pickup is selected first, where the name that is shown after the ‘on production line:’-text in figure 72 is the name that was given to the production line the Pally program node → Pickup-tab → Primary-tab (see section 6.2.1 Pickup).

Press the ‘Next’-button to select a pattern for the secondary pickup, the window shown in the figure below will then open. Note that the name of the production line is now set to the value

that was configured for the secondary pickup position in the Pally program node (see section 6.2.1 Pickup).

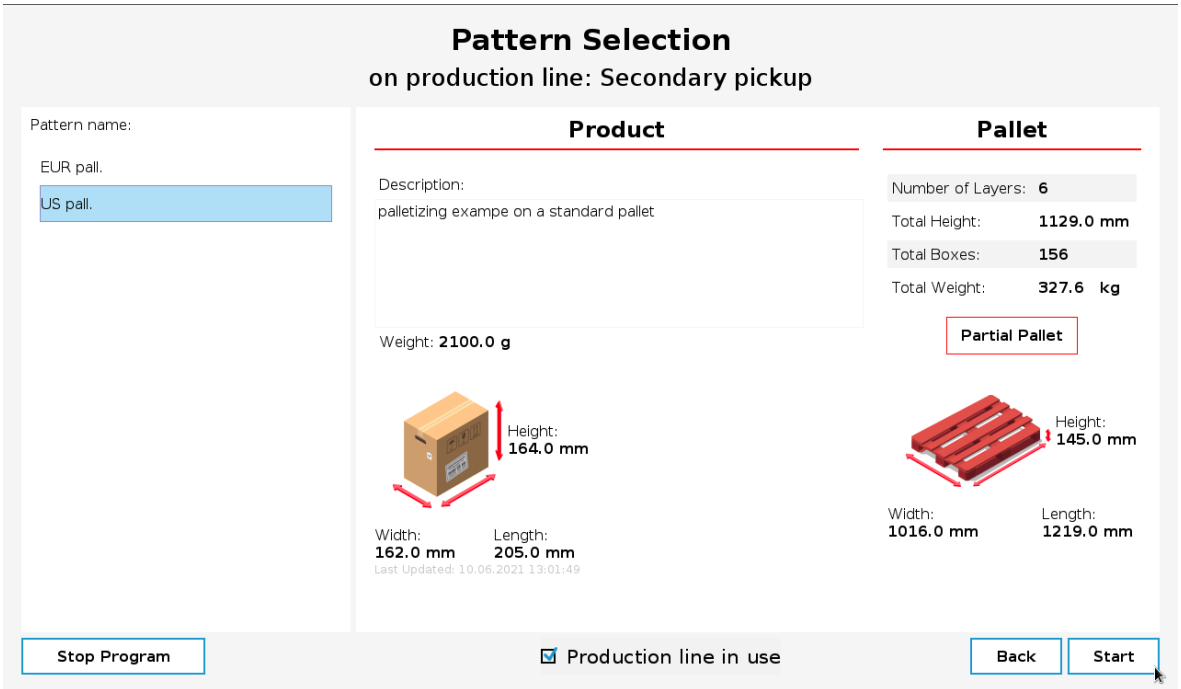


Figure 86: Pattern selection for the secondary pickup/production line in dual product mode

Select the desired pattern that is going to arrive on the secondary production line (i.e the line that is used for the secondary pickup position), then press the 'Start'-button to start palletizing. One of the three cases listed above (single pickup mode, dual pickup mode or dual product mode) will then be initiated, depending on the patterns and production lines that were selected.

Single Pickup Mode

If one of the production lines is not going to be used for a run of Pally, simply uncheck the 'Production line in use'-checkbox for that line. By disabling a production line, Pally will run in single *pickup* mode (1 product type from 1 pickup position). Please note that *at least* one production line has to be set to 'in use' for Pally to start.

Dual Pickup Mode

To achieve the case where Pally palletizes 1 product type from 2 pickup positions (i.e the dual *pickup* mode), then select the *same* pattern for the primary and secondary production lines and press the 'Start'-button. This will enable Pally to pick boxes from two production lines and place them on one pallet.

Dual Product Mode

To achieve the case where Pally palletizes 2 product types from 2 pickup positions (i.e the dual *product* mode), then select *different* patterns for the primary and secondary production lines and press the ‘Start’-button. This will enable Pally to pick boxes from two production lines and place them on two pallets. In practice, this means that Pally will place the two product types on *different* pallets, but the pallets will be filled with boxes *simultaneously*.

Note the nearest pallet is automatically assigned to each pickup position when palletizing two pallets from two pickup positions simultaneously.

Note if the same pattern is chosen for the two production lines, but one or more of the patterns have a partial pallet setting (as described in the next section), then Pally will run in dual *product* mode and not dual *pickup* mode.

Continue Existing Pallet(s)

If the program is stopped once Pally has palletized *at least* one box in any of the three cases (i.e single pickup mode, dual pickup mode or dual product mode), and the play button is pressed - then the ‘Continue existing pallet(s)’-window will appear. The look of this window will depend on the current case; either single pickup mode, dual pickup mode or dual product mode.

In single and dual pickup mode, the ‘Continue existing pallet’-window looks like the image below.

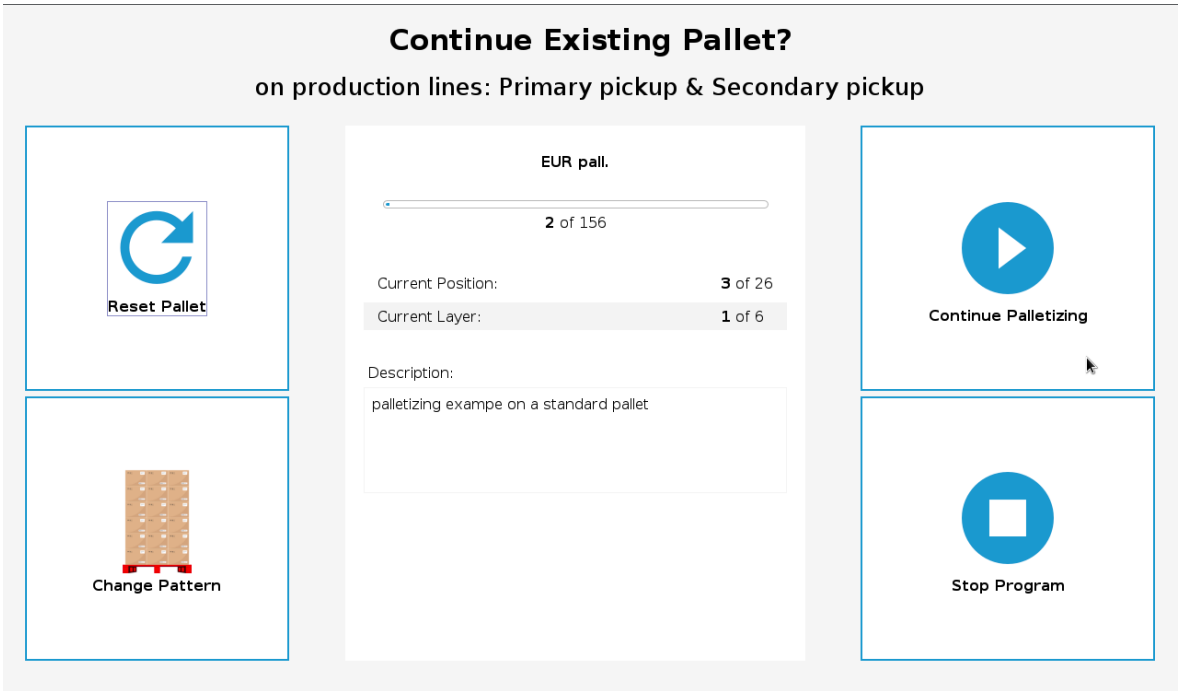


Figure 87: 'Continue existing pallet' window in dual pickup mode

If the program was running in a dual product mode state, i.e palletizing two different product types from two production lines; then the ‘Continue existing pallet’-window will look like the figure below. The progress of each pallet with its subsequent pattern is shown.

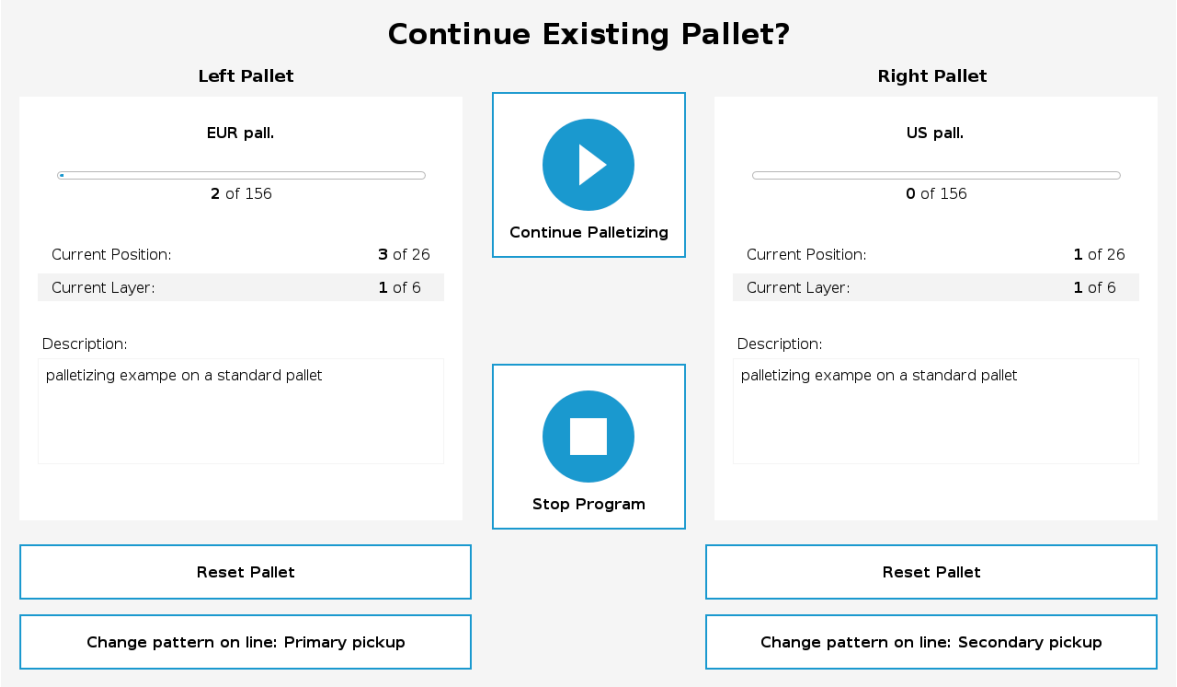


Figure 88: ‘Continue existing pallet’ window in dual product mode

It is possible to change the pattern on each production line (which subsequently changes the pattern on the pallet that is used by that production line) by pressing the ‘Change pattern on line: …’-button. If this button is pressed, then the pattern selection window for that production line is opened. This window is shown in the figure below.

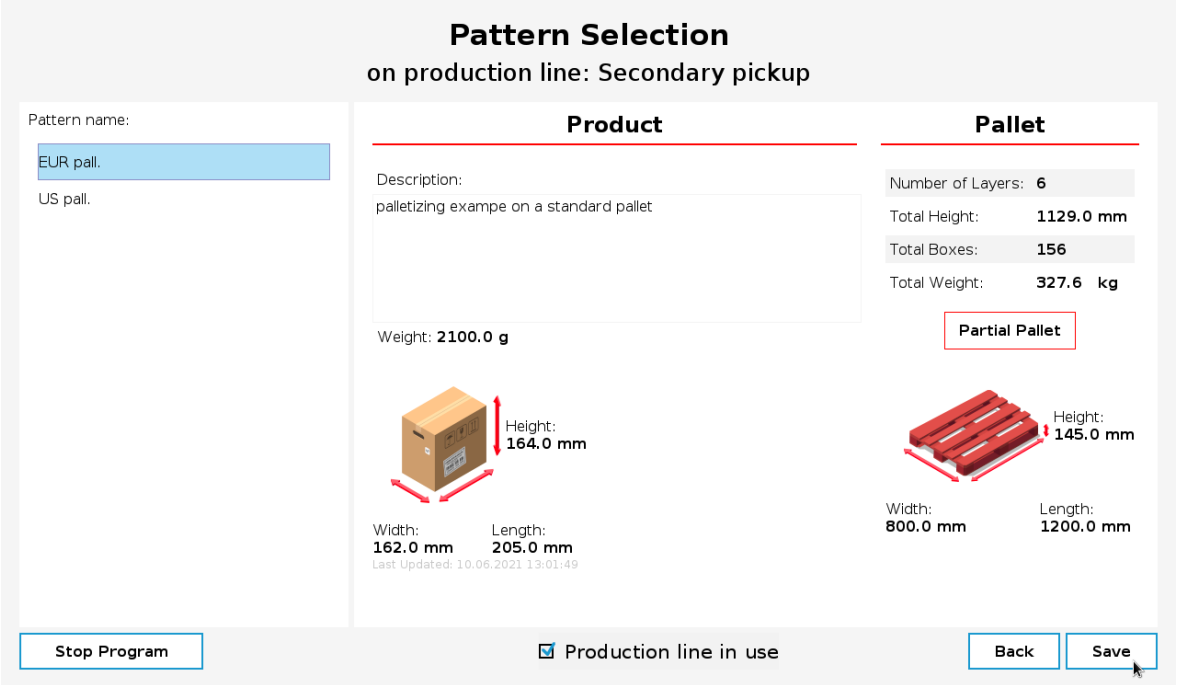


Figure 89: Pattern selection window when changing patterns on a specific line in dual product mode

To select a new pattern, and subsequently reset the pallet, select a pattern from the list and press the ‘Save’-button. Press ‘Back’ to keep the current pallet as is. It is also possible to stop using the production line by unchecking the ‘Production line in use’-checkbox.

7.3.3 - Partial Pallet

Note This feature only applies to the ***first*** pallet that is palletized. The remaining pallets will be palletized according to the selected pattern.

Note This description of the ‘Partial Pallet’-feature applies to both single and dual product mode (as described in 7.3.1 Single Product Mode and 7.3.2 Dual Product Mode).

To alter the start conditions on a pallet or the stop conditions, use the partial pallet feature. Once a pattern is selected in the ‘Pattern Selection’-dialog, a new dialog will open by clicking on the ‘Partial Pallet’ button.

In this dialog it is possible to edit two aspects of the selected pattern:

1. Edit the total boxes in the pattern
2. Edit the start conditions in the pattern

a. Edit the start layer
b. Edit the start box on the specified start layer

These two features are explained in more detail below.

7.3.3.1 Edit Total Boxes

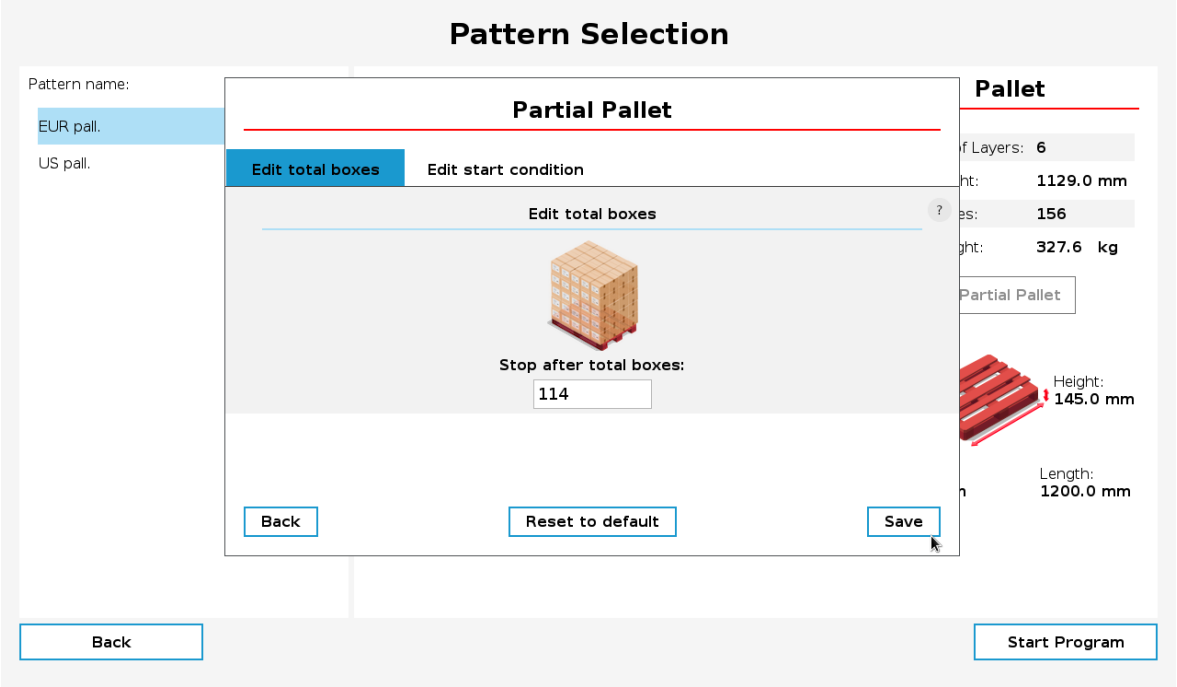


Figure 90: Edit the total boxes palletized on the first pallet

In order to palletize an incomplete pallet, type in the desired number of boxes in the input field under the ‘Stop after total boxes’-label. Make sure to input values in the range between 1 and the original amount of boxes in the selected pattern. Any total box number higher than the original amount of boxes will not be valid.

Note that the new value for the total boxes will only apply to the *first* pallet that is palletized, the remaining pallets will have the original amount of boxes in the selected pattern.

To reset the total boxes *and* the start conditions back to the default values for the selected pattern, press the ‘Reset to default’-button.

7.3.3.2 Edit Start Conditions

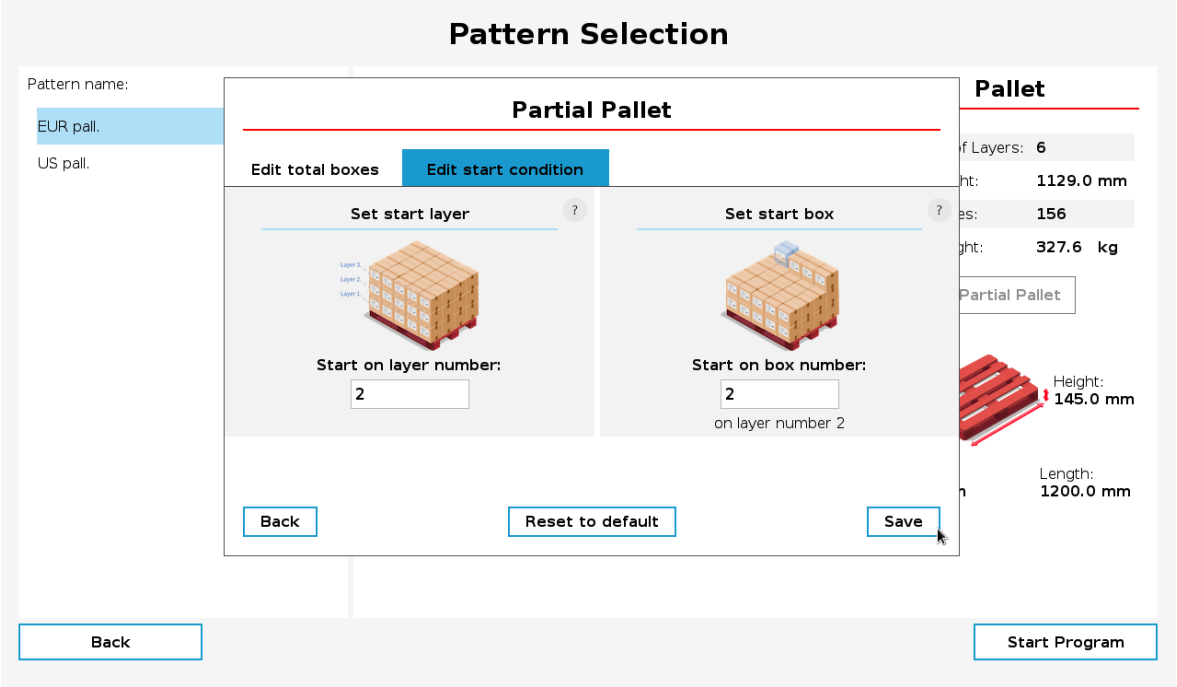


Figure 91: Edit the start conditions for the palletizer on the first pallet

In order to start palletizing from a different start condition than on the first box on the first layer, enter new values into the input fields below the ‘Start on box number’-label and the ‘Start on layer number’-label respectively.

The default values are 1 for both the new start layer and the new start box. Note that it is not possible to input values that are larger than the layer count for the selected pattern in the start layer input field. Be careful when entering a new start box, as there is no input validation on this field yet. One must therefore make sure that the number inserted into the new start box field is less than or equal to the amount of boxes that are present in the specified start layer.

If the selected pattern contains shim paper, it is important to note that these shim paper layers do **not** count as a layer. Therefore, to specify a new start layer for a pattern with shim paper(s), disregard each shim paper and specify the starting layer according to the layers containing boxes.

The new start box number signifies the box’s number on the selected layer, such that if the chosen start layer is the second layer and one wishes to start on the third box on the second layer, then input the number ‘3’ into the ‘Start on box number’-field.

7.3.4 - Order management (beta)

It is possible to palletize batches of partial pallets by creating an order list in advance.

7.3.4.1 Creating orders manually

To create a list of partial pallets, select "Partial Pallet" and go to the "Order management (beta)" tab.
Enter the number of boxes to be palletized, and press the "Add to order" button. If the value exceeds the total number of boxes in a full pallet, the program will split the order into multiple pallets.
You can add as many full and/or partial pallets as needed.
The left side of the window shows the list of pallets to be palletized with the corresponding number of boxes. To remove a pallet, select the entry in the list and press "Remove from order".
The program will complete the specified pallets and then terminate. To begin with a new order, it is necessary to restart the program.

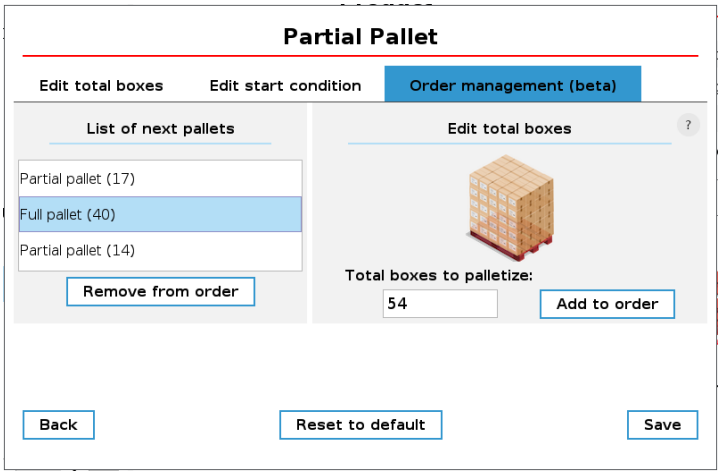


Figure 92: Create a list of full and partial pallets in advance

7.3.4.2 Creating orders remotely

It is possible to create orders via Pallet Manager Daemon API functions. This is especially useful in projects where the robot is controlled by a PLC or another external device.
The program can run in the following 2 operational modes:
rf_order_mode = 1: complete the orders and terminate
rf_order_mode = 2: complete the orders and wait for a new order

---

Note In this automated mode it is possible to create subsequent orders with different patterns. The program will proceed to the next order regardless of the pattern (and product) selected. The operators must ensure the correct boxes are provided on the conveyor before the program starts processing the order.

7.4 - Recovery from error

When the program stops and must be started again, remove any boxes from the gripper and turn off the vacuum. Use the freedrive button to move the robot into a position that is close enough to the default wait position, so the robot can start without collision. It may be necessary to set the empty gripper weight as current payload manually before activating the robot arm.

7.5 - Pally Operator Panel (POP)

When the operator interface type is set to "Complete", a progress indicator window appears during palletizing. The operator can confirm the left and right pallet positions and get an overview of the current palletizing status, including the name of the pattern, current layer and box position, elapsed time, estimated remaining time, average CPM and cycle time, previous pallet time, etc.
By pressing the red X icon in the pallet status area, the program stops palletizing on the current pallet, and after a new pallet confirmation it starts a new empty pallet.
The pause and stop buttons work identical to the default UR pause and stop buttons.

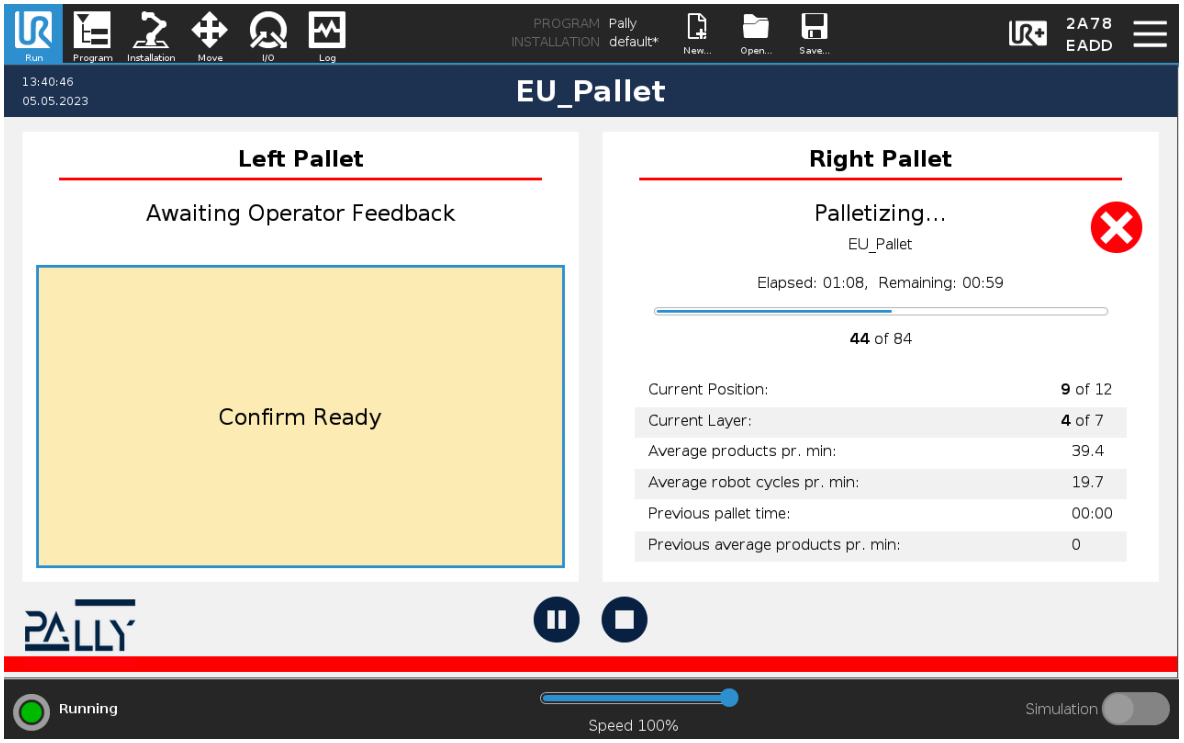


Figure 93: the Operator Panel during palletizing

Note: The Operator Panel disappears automatically when the robot stops with protective stop or emergency stop, and when clicking outside the operator panel window.

Note: Detailed palletizing status view is not available without Service and Support plan.

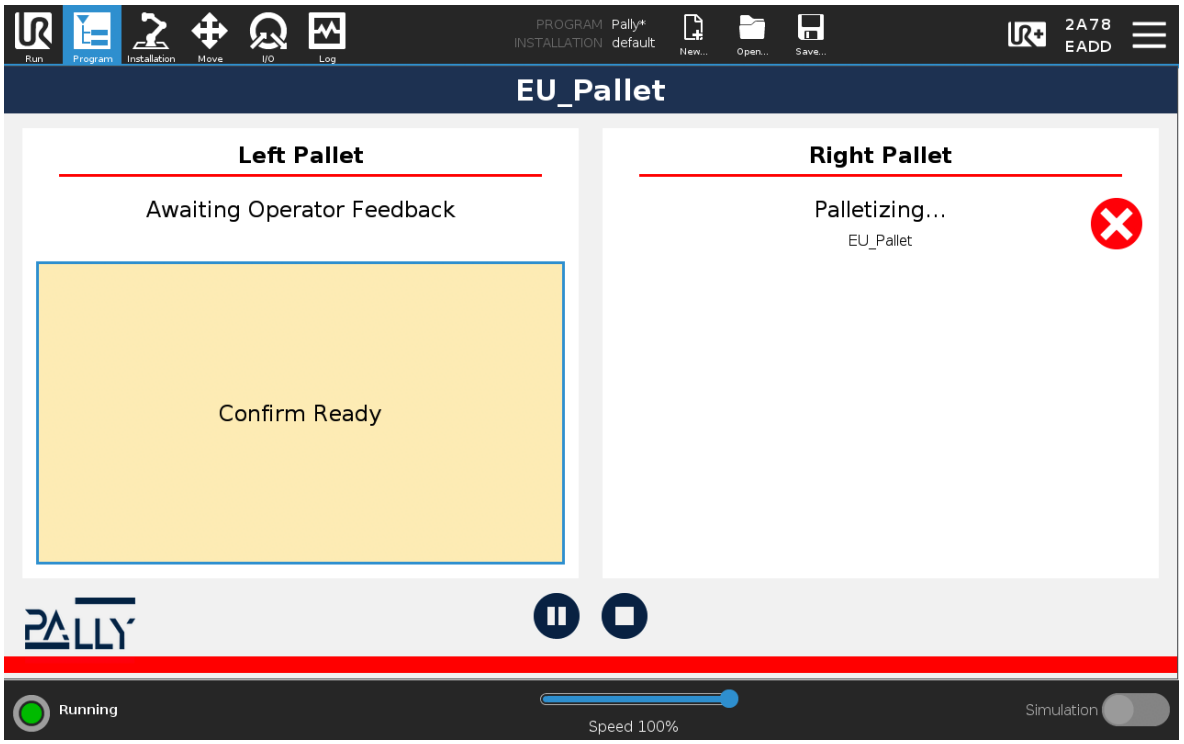


Figure 94: the Operator Panel has limited functionality with expired service plan

7.6 - UR+ toolbar

Use the UR+ toolbar button to restore the hidden operator interface of Pally while the program is running.

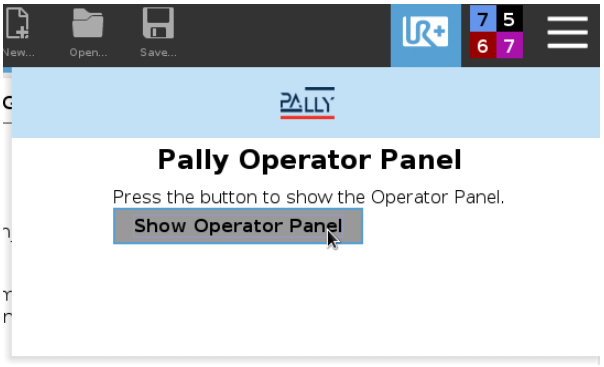


Figure 95: UR+ toolbar button to restore the operator interface

---

Note: a functionally equivalent button is also available on the CB-series Polyscope 3.x.

7.7 - Show mode

When starting the program in Show Mode, it does not include the usual product selection screen. The pattern name needs to be given in the “rf_product_selection_predefined” variable, as described in the Automation section.
Show Mode has a simplified User Interface where the operator can select the start conditions for the demo program by selecting one of the given options:

- Continue from the last position where the program previously stopped
- Left pallet is full, or Right pallet is full
- Operator counts the number of boxes on the Left pallet or Right pallet
- Build an initial Left pallet or Right pallet first, then proceed with the demo
- Show the Product Selection GUI only

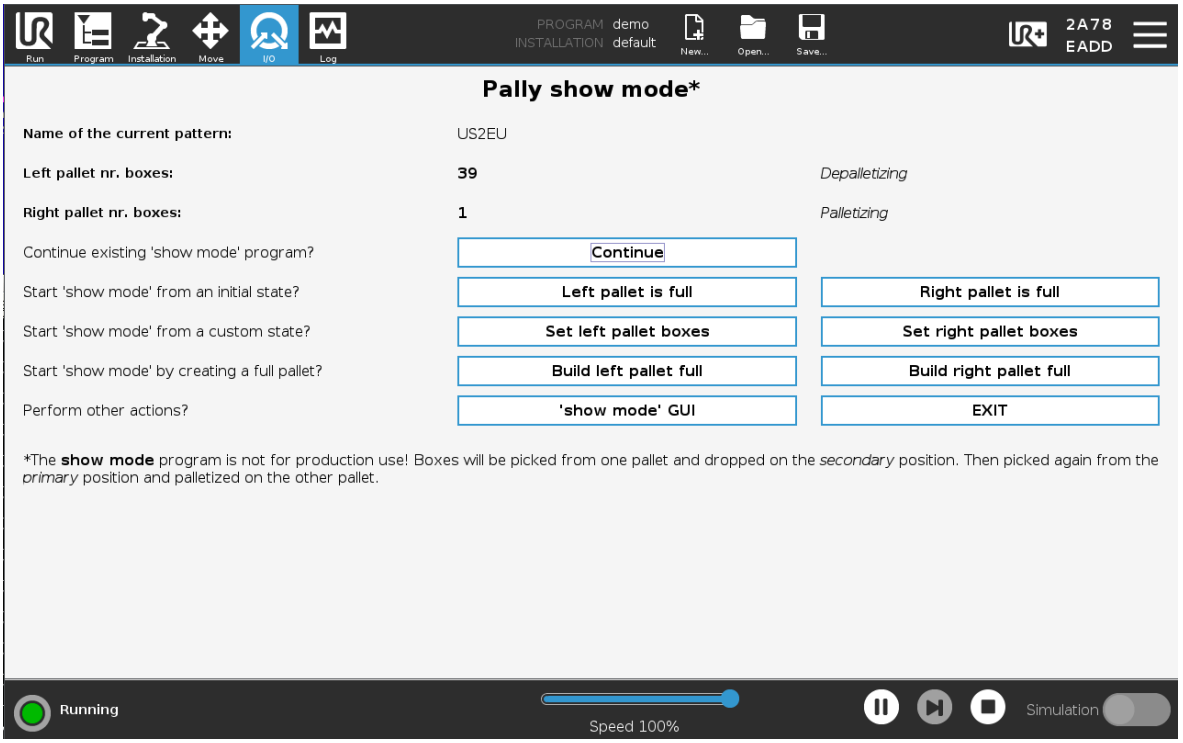


Figure 96: Show mode

8 - Creating Patterns

This chapter describes the pallet patterns in detail, and gives an overview of the data structures used.
To start defining patterns, use a web browser and navigate to https://palletizer.rocketfarm.no where you can create your patterns in a graphical editing environment. All patterns can be exported to a regular text-file and further edited in any text-editor.

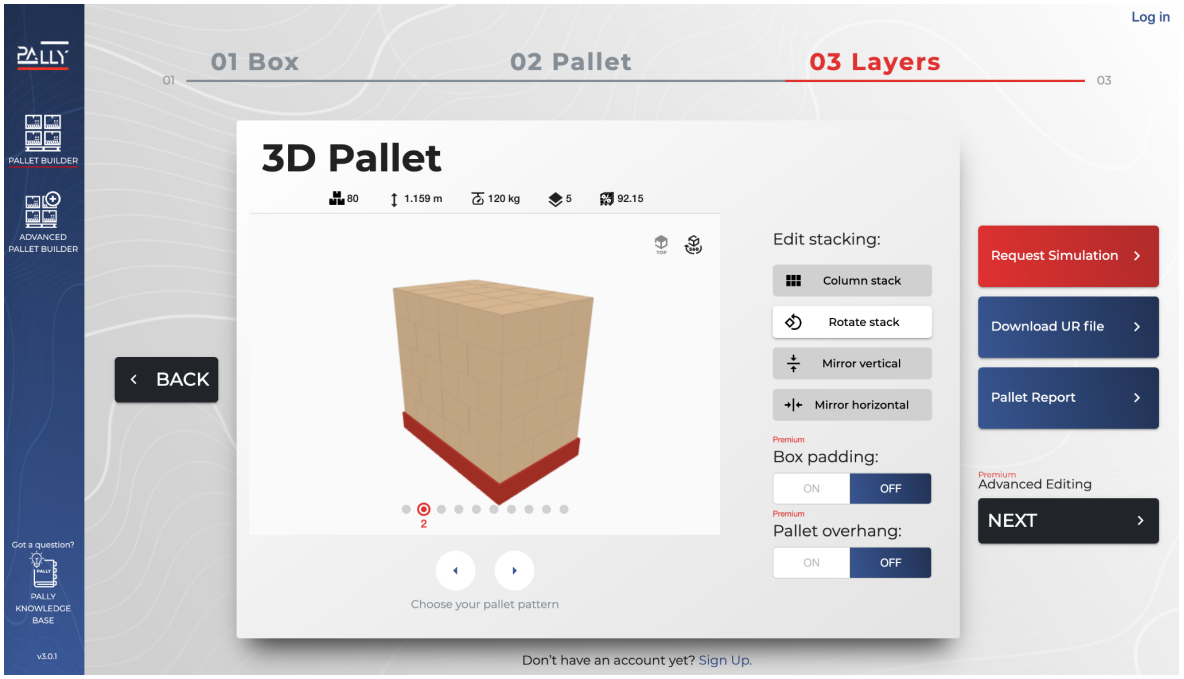


Figure 97: the graphical pattern editing environment at https://palletizer.rocketfarm.no

8.1 - The pallet pattern definition file

Each product type requires its own pallet pattern definition file. This file includes all product-specific information that the robot needs for palletizing:

- product data (see Project overview)
- layer types (see Layer types)
- layers (see Layers)
- zones (see Zones)

8.2 - Project overview

This section of the pattern file includes base data about the product itself, such as

- the dimensions (width, length, height) of the pallet, mm:
- height of the empty pallet (see palletHeight), mm:
- the dimensions (width, length, height) of one box of product, mm:
- the weight of one box, g:
- the label position on the box (labelOrientation):
    - 0: label in front
    - -90: label on the left side
    - 90: label on the right side
    - 180: label on the back side
    - null or missing value: label not defined
- the maximum number of boxes the robot can pick at once (maxGrip)
    - 1: single pick

    - 2: double pick
    - 3: triple pick
    - … (up to 8-pick) or
- automatic calculation of the maximum number of boxes in multi-grip (maxGripAuto)
- a unique name, a description, and a timestamp,
- and some additional metadata for the graphical pattern editor.

It is important that the product dimensions are as accurate as possible to achieve the best possible result.

```
{
    "name": "test",
    "description": "my first pattern",
    "dateModified": "2020-04-29T07:03:20.288Z",
    "dimensions": {
        "height": 1048,
        "length": 1200,
        "width": 800,
        "palletHeight": 145
    },
    "productDimensions": {
        "width": 365,
        "height": 262,
        "length": 377,
        "weight": 5000
    },
    "maxGrip": 1,
    "maxGripAuto": true,
    "labelOrientation": 0,
```

Figure 98: the project overview

8.3 - Layer types

Layer types define possible layouts of individual layers. Depending on the desired stacking method, the pallet can be built by using the same layer type repeatedly (column stack) or alternating two or more different layouts (flipped, rotated, etc.) in order to create interlocking between the layers for increased stability.

```
{
    [...]
    "layerTypes": [
        {
            "name": "normal",
            "pattern": [
```

```
            {
                "x": 588.5,
                "y": 217,
                "r": [90, 270]
            },
            {
                "x": 211.5,
                "y": 217,
                "r": [90, 270],
                "g": [0],
                "f": 1

            },
            [...]
        ],
        "altPattern": [
            {
                "x": 588.5,
                "y": 217,
                "r": [90, 270]
            },
            [...]
        ],
        "approach": "inverse"
    },
    {
        "name": "rotated",
        "pattern": [
            {
                [...]
```

Figure 99: layer types

8.3.1 - Box position

Each box has a specific position (the "x" and "y" values) relative to the pallet corner. The edges of the pallet define a Cartesian coordinate-system with the origo in the lower left corner. By default, positions are defined in millimeters.

Note: The layer type is a 2-dimensional logical model of the layout, thus only the x and y coordinates are defined. The z-coordinate will be calculated from the product height and the actual layer being palletized. The [x, y, z] values point to the center point at the top of the box on the pallet.

8.3.2 - Box rotation

Each product has at least one allowed rotation (the "r" values) on the pallet, which can be 0, 90, 180, or 270 degrees, or the combinations of [0, 180] and [90, 270] respectively.

8.3.3 - Box order

Boxes are palletized in the same order as they are defined in the pattern file. All positions of one layer are completed before the robot continues on the next layer. The only exception is the use of zones, see Zones.

8.3.4 - Enforced single/multi-grip

By default, the robot will try to palletize two or more adjacent boxes in one turn, in order to maximize throughput. The number of boxes that the robot can pick at once is limited by the maxGrip parameter in the pattern file as well as the installed product sensors on the production line and the size of the gripper.
In order to force the robot to pick *less than* the maximum possible number of boxes, the separator flag (the "f" value) can be used. The separator flag will instruct the robot to stop looking for more boxes. An “f”-value set to 1 means that the robot will check to see if it can pick the current box with the next box in the pattern. If the “f”-value is set to 0, then the robot will *not* try to pick the current box with the *next* box.

The following outtake of a pattern file shows how it is possible to force the robot to complete a double-pick followed by a single-pick for the three first boxes in a pattern where the robot can pick three boxes at once.

```
"maxGrip": 3
[...]
      "pattern": [
                        {
                            "x": 690,
                            "y": 165,
                            "r": [0, 180],
                            "f": 1
                        },
                        {
                            "x": 690,
                            "y": 455,
                            "r": [0, 180],
                            "f": 0
                        },
                        {
                            "x": 690,
                            "y": 745,
                            "r": [0, 180],
                            "f": 0
```

```
                        },
                        [...]
                    ],
[...]
```

Figure 100: positions in a layer type

8.3.5 - Gripper rotation at pickup

By default, the robot will choose the best gripper position when picking up the box(es) from the conveyor. The possible options are specified by the gripper optimization settings in the URCap (4-ways, 2-ways, or none).
For specific box positions however, it is possible to enforce a gripper position (the "g" values) or limit the possible gripper positions the robot can choose from. Any combinations of the following values are allowed: 0, 90, 180, and 270, where 0 corresponds to the configured pickup position in the URCap, other values are angles, counterclockwise when seen from above.

8.3.6 - Left and right pallet layout

Each layout can be defined separately for the left and right pallet position. Even if the left and right pallets look identical, the order of the boxes can be different.
The default layout (the "pattern" structure) defines the layout for the right pallet. The alternative layout (the "altPattern" structure) defines the layout for the left pallet. By default - if no alternative layout is defined - the robot will generate a mirrored layout for the left pallet, in order to make a symmetric setup.

8.3.7 - Inverse approach

For each layer type it is possible to define how the robot should approach the target position (the "approach" value corresponds to the right, and the "altApproach" to the left pallet layout). The approach direction should depend on the box order - palletizing from the furthest side of the pallet and in ("normal") or starting at the nearest side and continuing further out ("inverse"). In special circumstances - especially when the mounting base is wide - the inverse box order and approach produces a better result with less or no collision situations.

8.3.8 - Shim paper

Shim papers are considered as special layers without actual products. The shim paper layer has its own height, usually 2-3 mm. The number of shim paper layers is not limited.
The shim paper layer type is marked with the "class": "separator" field and must not contain the "pattern" and "altPattern" structures.

```
    "layerTypes": [
```

```
        {
            "name": "shimPaper",
            "class": "separator",
            "height": 3
        } ...
    ],
    "layers": [
        "shimPaper", ...
```

Figure 101: definition and usage of a shim paper layer type

8.4 - Layers

The "layers" structure defines the actual layout of the final pallet. Each layer is built according to one of the layer types as described above.

```
{
    [...]
    "layers": [
        "normal",
        "rotated",
        "shimPaper",
        "normal",
        "rotated"
    ]
```

Figure 102: layers

Note: The first element in the layer list is the first layer on the empty pallet. The last element in the list is the topmost layer.

8.5 - Zones

Normally the robot completes a layer before starting the next layer. However, the pallet can be split into different regions (the "zones" and the optional "altZones" structure) which override the regular layer completion rules. In general, a zone cannot be started until the previous zone is finished.
Zones can divide the pallet along the X, Y, and Z-axis. For example, it is possible to complete all layers of the furthest (outer) half of the pallet before proceeding to the nearest (inner) half.
The URCap allows inserting extra program code before starting and after completing a zone, in order to communicate with any optional external hardware or software or perform custom operation when a specific part of the pallet is started and/or completed.

Each zone definition can contain a position transformation hint (the "posData" array) which may be used by the Pally URCap to control a lifting column. The posData value is optional, and the program will not control the lifting column unless the value is explicitly defined.

```
{ [...]
    "zones" : [
        {
            "z" : { "condition" : "<=", "value" : 730 },
            "posData" : [0,0,0]
        },
        {
            "z" : { "condition" : "<=", "value" : 980 },
            "posData" : [0,0,466]
        },
        {
            "x" : { "condition" : ">=", "value" : 200 },
            "z" : { "condition" : "<=", "value" : 1220 },
            "posData" : [0,0,466]
        },
        {
            "x" : { "condition" : ">=", "value" : 500 },
            "z" : { "condition" : ">=", "value" : 1220 },
            "posData" : [0,0,466]
        },
        {
            "x" : { "condition" : "<=", "value" : 200 },
            "z" : { "condition" : "<=", "value" : 1220 },
            "posData" : [0,0,466]
        },
        {
            "x" : { "condition" : "<=", "value" : 500 },
            "z" : { "condition" : ">=", "value" : 1220 },
            "posData" : [0,0,466]
        }
    ],
    "altZones" : [
        {
            [...]
```

FigureFigure 103: zones in the pattern file

8.5.1 - Conditions

Zones are defined by using relational operators "<", "<=", ">", ">=" on the x, y, and z axis in the Cartesian coordinate-system defined by the pallet edges with the origo in the lower left corner of the pallet. If multiple conditions are used, they have to be enclosed in an array:

```
"x" : [
      {"condition" : ">", "value": 500} ,
      {"condition" : "<=", "value" : 800} ]
```

Figure 104: multiple conditions in a zone definition

Note: Zones are evaluated in the order they are defined. If two or more zones are overlapping, the program will choose the first one that matches the condition criteria. This might cause some zones to never be activated.

8.5.2 - Position transformation

Each zone can define a position transformation hint, which may be used by the Pally URCap to control a lifting column and/or a base slider. Before the robot starts palletizing the specified zone, the external hardware is controlled by the Pally program to put the robot into the desired mounting position.
The "posData" value has the following format: [x, y, z] in the robot base coordinate system (in millimeters). The default position is [0, 0, 0].

8.5.3 - Location of the zone definitions

For compatibility reasons it is possible to define the zones in 3 different locations, which are loaded in the following order:

1. the "zones" and "altZones" structures included in the pattern file (recommended),
2. a separate file called *pattern*.zones.json - for a pattern file called *pattern*.json
3. a default file called zones.json
4. automatically generated by the Pally URCap for each product (only when a supported lifting column is installed and configured properly)

8.6 - Uploading new patterns

Place the JSON-files containing the patterns on the root folder of a USB-stick and plug this USB-stick into the robot. Depending on the settings for the pattern upload that is enabled in the ‘Advanced’ → ‘Patterns’ sub-tab (see section 6.1.6 Advanced), the JSON-files are either automatically uploaded to the desired storage directory on the robot or the ‘Upload Patterns’ button in the ‘Patterns’ sub-tab must be pressed to copy the JSON-files onto the robot.

In addition, as the JSON-files are uploaded onto the robot, all existing json files from the robot will be backed up in a folder labeled ‘backup.{date}’ on the supplied USB-stick.

9 - Advanced configuration variables

All settings on the Pally graphical configuration panels are translated into URScript local variables, which - along with some additional variables - determine the final behavior of the palletizer program. To get better control of the program, or change some settings in runtime, it may be necessary to alter one or more of these configuration variables.

9.1 - Naming conventions

All configuration variables have names beginning with "rf_". The variables are visible only under the Pally program node, but can be enforced to be globally visible by declaring global variables with the same name above the Pally program node in the program structure.

9.2 - Changing configuration variables at startup

In this scenario, the configuration variables are changed once at startup, before entering the main loop of the palletizer program. To change any configuration variable, use the initial MoveJ program node under the Pally program node to enter script lines or script files that modify the variables. The global keyword is not required in this case.

9.3 - Changing configuration at runtime

It might be necessary to modify some configuration variables from a Pally callback, e.g. change the approach distance for some specific products only. Due to limitations in the URScript program structure, the rf-variables are not visible in the callbacks. In this case it is necessary to enforce the corresponding variable to be global.

Note: Changing some configuration variables such as the pallet calibration points in runtime won't have any effect on the current pallet as the values are only used once after product selection.

9.4 - List of available configuration variables

The following tables contain a short explanation of the rf-variables with their data source. Related variables are categorized into separate tables.

9.4.1 - Speed and acceleration

Name	Description	Reference

rf_max_acceleration	Maximum acceleration	Program / Movement / Speed / Acceleration
rf_speed	Maximum tool speed	Program / Movement / Speed / Speed
rf_precise_acceleration	Approach acceleration	Program / Movement / Speed / Approach Acceleration
rf_precise_speed	Approach tool speed	Program / Movement / Speed / Approach Speed
rf_min_grip_quality	Minimum required grip quality for moving with full acceleration	Program / Movement / Speed / Smart acceleration

9.4.2 - Calibration points

Name	Description	Reference
rf_p11	Right pallet lower right calibration point	Program / Pallet / Right
rf_p13	Right pallet upper left calibration point	Program / Pallet / Right
rf_p14	Right pallet lower left calibration point	Program / Pallet / Right
rf_p21	Left pallet lower left calibration point	Program / Pallet / Left
rf_p23	Left pallet upper right calibration point	Program / Pallet / Left
rf_p24	Left pallet lower right calibration point	Program / Pallet / Left
rf_boxpickup	Primary pickup calibration point	Program / Pickup / Primary
rf_con1_w	Primary pickup conveyor total width	Program / Pickup / Primary
rf_guide1_w	Primary pickup fixed guide width	Program / Pickup / Primary
rf_guide1_dir	Primary pickup fixed guide position	Program / Pickup / Primary

rf_boxpickup_2	Secondary pickup calibration point	Program / Pickup / Secondary
rf_con2_w	Secondary pickup conveyor total width	Program / Pickup / Secondary
rf_guide2_w	Secondary pickup fixed guide width	Program / Pickup / Secondary
rf_guide2_dir	Secondary pickup fixed guide position	Program / Pickup / Secondary
rf_calib_pu_dim	Calibration box dimensions	Program / Pickup / Calibration box
rf_calib_box_dim	Alternative calibration box dimensions for pallet calibration	Default: identical to rf_calib_pu_dim
rf_P1_calib_h	Height of the pallet that was used during calibration of the right pallet position	Program / Pallet / Height
rf_P2_calib_h	Height of the pallet that was used during calibration of the left pallet position	Identical to rf_P1_calib_h

9.4.3 - Path planning

Name	Description	Reference
rf_above_pickup	Distance from pickup to above pickup position	Program / Advanced / Path planning
rf_box_free	Distance from pickup to box free position	Program / Movement / Waypoints / Box free height
rf_smart_exit_p1_min_x	Right pallet smart exit X lower limit	Program / Advanced / Path planning
rf_smart_exit_p1_max_x	Right pallet smart exit X upper limit	Program / Advanced / Path planning
rf_smart_exit_p1_min_y	Right pallet smart exit Y lower limit	Program / Advanced / Path planning
rf_smart_exit_p1_max_y	Right pallet smart exit	Program / Advanced / Path

	Y upper limit	planning
rf_smart_exit_p2_min_x	Left pallet smart exit X lower limit	Program / Advanced / Path planning
rf_smart_exit_p2_max_x	Left pallet smart exit X upper limit	Program / Advanced / Path planning
rf_smart_exit_p2_min_y	Left pallet smart exit Y lower limit	Program / Advanced / Path planning
rf_smart_exit_p2_max_y	Left pallet smart exit Y upper limit	Program / Advanced / Path planning
rf_pallet_edge	Pallet lip	Program / Movement / Waypoints / Pallet lip
rf_approach	Approach distance	Program / Movement / Waypoints / Approach
rf_high_appr_boost	High approach boost	Program / Advanced / Path planning / High approach boost
rf_collision_radius_min	Collision radius (in meters) around the Z axis when moving above the top of the shoulder joint	Default: 0.15
rf_collision_radius_max	Collision radius around the Z axis when moving below the top of the shoulder joint	Default: 0.35
rf_approach_auto	Approach distance can be recalculated for each box in run-time	Program / Movement / Waypoints / Approach Distance / Fixed Default: True (checkbox 'fixed' is NOT selected)
rf_box_free_auto	Box free height should not be smaller than the box height.	Program / Movement / Waypoints / Box Free / Fixed Default: False (checkbox 'fixed' is selected)
rf_quick_return	Enable the Return path Optimizer for 2-ways and 4-ways gripper optimization and Dual Product mode. May improve performance.	Default: True

9.4.4 - Gripper

Name	Description	Reference
rf_gripper_width	Gripper width in meters	Installation / Gripper / Dimensions
rf_gripper_length	Gripper length	Installation / Gripper / Dimensions
rf_gripper_height	Gripper height	Installation / Gripper / Dimensions
rf_gripper_weight	Gripper weight in kg	Installation / Gripper / Dimensions
rf_grip_rot	Gripper optimization	Program / Advanced / Path planning / Optimize gripper at pickup
rf_max_grip	Maximum number of boxes in multi-pick	Default: number of product-sensors installed
rf_max_grip_line	Maximum number of boxes in multi-pick for each conveyor - in Dual Product Mode	Default: number of product-sensors installed on each conveyor [l1, l2]
rf_grip_delay	Delay after turning on the gripper at pickup, in seconds	Default: 0.2
rf_grip_release_delay	Delay after turning off the gripper at target	Default: 0.2
rf_foam_height	Gripper foam (suction cup) squeezing length in meters	Default: 0.02
rf_grip_rot_cost	Additional cost of each specific gripper rotation (0, 180, -90, +90 degrees in this order)	Default: [0, 30, 10, 10]
rf_length_coverage	Minimum required fraction of the total length of each box covered by the gripper in multi-pick	Default: 0.2

9.4.5 - Lifting column

Name	Description	Reference

rf_lift_max_stroke	Maximum stroke in meters	Installation / Lifting column / Characteristics / Maximum Stroke
rf_lift_min_stroke	Minimum stroke (can be negative as well)	Default: 0
rf_lift_wait_time	Maximum time to wait for lifting column (in seconds)	Installation / Lifting column / Characteristics / Maximum Duration
rf_lift_safe_up	Fraction of the total stroke that is reserved at the top as a safety margin in dynamic positioning calculations.	Default: 0.05
rf_lift_safe_down	Fraction of the total stroke that is reserved at the bottom as a safety margin in dynamic positioning calculations.	Default: 0.05
rf_lift_alt_boost	Preferred lifting column position as a fractional number between the lowest and highest possible positions.	Default: 1 (keep as high as possible)
rf_lift_optimistic	Dynamic Positioning will try to re-use the last used lifting column position whenever possible	Default: True (re-use current lifting column position whenever possible)
rf_lift_alt_boost_auto	Instructs the optimizer to overwrite rf_lift_alt_boost and rf_lift_safe_down for the given pattern.	Default: True (initialize rf_lift_alt_boost and rf_lift_safe_down parameters for the specific pattern and mode) - the user can still update these values in the onNextTask callback.

9.4.6 - Automation

Name	Description	Reference
rf_product_selection_strategy	Method to select product on startup: 0: GUI	Default: 0

	1: First 2: Predefined	
rf_product_selection_predefined	Name of the pattern file - without the file extension ".json" - when product selection strategy is 2. In Dual Product mode: two pattern names for the primary and secondary conveyor should be given in the following format: "pattern1;:;pattern2" (Use separator ";:;" between names)	
rf_P1_terminate	Force finish existing Right pallet on startup	Default: False
rf_P2_terminate	Force finish existing Left pallet on startup	Default: False

9.4.7 - Other/Special

Name	Description	Reference
rf_goto_wait	Go through the default waiting position before picking the next box	Default: False
rf_release_strategy	Bitmask that selects one or more waypoints to be removed from the return path Bit 0: remove box free Bit 1: remove target position Bit 2: use vertical exit path Bit 3: remove approach point Bit 4: remove everything after smart-exit point	Default: 7 (remove box free, remove target position, use vertical exit)
rf_custom_path	Skip path planning completely (use in combination with user-defined path)	Default: False
rf_blend	Blend radius in linear movements as a	Default: 0.3 Recommended values: 0 - 0.3

	fraction of the distance between subsequent waypoints.	
rf_safe_reach	~~Workaround for a UR bug where the inverse kinematics calculations may throw an exception.~~ Deprecated and no longer used.	Default: 0.01
rf_use_movej	The robot uses joint move (movej) to approach the pick position. Use this option when experiencing "Joint position close to limits" due to wrist 3 winding up during palletizing.	Default: True
rf_cpuidle_ticks	Threshold value for inserting idle instructions to avoid "Runtime is too much behind" and "Lost communication with controller" issues.	Default: 20 on E-series Default: 100000 on CB-series Has no effect on CB 3.0
rf_cpuidle_l	Increment for the idle tick counter, variant 1	Default: 2 on E-series Default: 1 on CB-series Has no effect on CB 3.0
rf_cpuidle_xl	Increment for the idle tick counter, variant 2	Default: 4 on E-series Default: 1 on CB-series Has no effect on CB 3.0
rf_cpuidle_xxl	Increment for the idle tick counter, variant 3	Default: 25 on E-series Default: 20000 on CB-series Has no effect on CB 3.0
rf_ur_wd_timeout	Workaround for a UR bug where a move command never terminates	Default: 60 (seconds)
rf_joint_deviation_max	Maximum allowed difference (in radians) between actual joint positions and target	Default: 0.0043 (in radians) See Payload control and robot stabilization for further information.

	joint positions while the gripper is being operated during pickup and release. Setting to 0 will disable joint deviation monitoring at this point.	
rf_joint_deviation_err	Maximum allowed difference (in radians) between actual joint positions and target joint positions before turning on the gripper for picking the box. Setting to 0 will disable joint deviation monitoring at this point.	Default: 0.0001 (in radians) See Payload control and robot stabilization for further information.
rf_final_blend_radius	Blend radius between the last waypoint on the return path and the first waypoint towards the next pickup. NOTE: Blend between moveL and moveJ may often cause unpredictable protective stops.	Default: 0 Recommended: between 0 - 0.01
rf_max_payload	Maximum payload suitable for this robot (in kg.) incl. gripper	Derived from robot serial number. (3, 5, 10, 12.5, 16, 20 kg depending on the robot model)

10 - PalletManager daemon process

The Pallet Manager daemon is a process that runs in the background, making it possible to keep the pallet completion state when the robot program is stopped and continue from the same position when the program is started again.
In this chapter you will find the most common functions of the daemon that are useful for advanced palletizing solutions. However, not all functions are listed here.

10.1 - Using the PalletManager daemon

A variable called `Pally_daemon` is automatically created on startup, and is available in the entire program scope. For compatibility reasons, the variable `palletmanager_daemon` is provided locally under the callbacks and in the initial MoveJ program node under the Pally program node.
Methods of the daemon can be invoked from URScript code in the following format:

```
result = palletmanager_daemon.method_name(param1, param2…)
```

Where method_name is one of the following methods listed below.

10.2 - Pallet dimensions

To obtain the pallet dimensions from the currently loaded JSON pattern file, use the following method:

```
palletDim = palletmanager_daemon.get_pallet_dimensions(0)
```

The method has one input parameter that is reserved and must be 0 in typical setups.
The result is an array with 3 elements: width, length, and height, in millimeters. The following example converts each value to meters and stores them in 3 separate variables:

```
palletWidth = palletDim[0] / 1000
palletLength = palletDim[1] / 1000
palletHeight = palletDim[2] / 1000
```

10.3 - Product dimensions

To obtain the product dimensions from the currently loaded JSON pattern file, use the following method:

```
productDim = palletmanager_daemon.get_product_dimensions(0)
```

The method has one input parameter that is reserved and must be 0 in typical setups.
The result is an array with 4 elements: width, length, and height, in millimeters, and weight in grams. The following example converts each value to meters and kilogram, and stores them in 4 separate variables:

```
productWidth = productDim[0] / 1000
productLength = productDim[1] / 1000
```

```
productHeight = productDim[2] / 1000
productWeight = productDim[3] / 1000
```

10.4 - Pallet completion state

To obtain the current pallet completion state (progress indicator) use the following method:

```
palletState = palletmanager_daemon.get_pallet_state(0)
```

The method has one input parameter that is reserved and must be 0 in typical setups.
The result is an array with 3 elements: total number of boxes, number of boxes completed, number of boxes not completed.
Please note that boxes in the current task are not included in the completed and the uncompleted boxes.
Use the following example to get detailed status of the pallet:

```
palletState = palletmanager_daemon.get_pallet_state(0)
nrTotal = palletState[0] # total number of boxes
nrDone = palletState[1] # completed
nrNotDone = palletState[2] # not completed
nrCurrent = nrTotal - nrDone - nrNotDone # being completed now
isSinglePick = (1==nrCurrent) # boolean value, True for single
pick
```

10.5 - Instance parameter in Dual Product mode

In Dual Product mode, the program keeps track of two separate pallet completion states in memory, called 'instances'. That introduces a parameter 'instance' in several Pallet Manager daemon functions. Instance is usually the first parameter and is always 0 in the previous examples.

```
palletStatePrimaryLine = palletmanager_daemon.get_pallet_state(0)
palletStateSecondaryLine =
palletmanager_daemon.get_pallet_state(1)
```

Instances are assigned to pick positions, not pallet positions: the primary line belongs to instance 0 and the secondary to 1. Which instance belongs to the left and right pallet can be derived from which pallet is assigned to each pick position (shortest distance from the corresponding conveyor).

11 - Troubleshooting

In this chapter you will find the most common questions and answers about the Pally urcap.

	Symptom	Solution
1	Unexpected safety stop before pickup	Move the calibrated pickup position higher Check product height
2	Unexpected safety stop after pickup	Check product weight Check vacuum Adjust the **rf_grip_delay** value
3	Unexpected safety stop during motion between pickup and pallet positions	Use a smaller Maximum Acceleration value Adjust the Smart Acceleration value Reduce the maximum allowed speed of the base joint in the robot Safety settings
4	Unexpected safety stop before release	Move the calibrated pallet position higher Check product height
5	Unexpected safety stop after release	Check product weight Adjust the **rf_grip_release_delay** value
6	Box collision to the conveyor	Check conveyor dimension settings Check fixed guide position (left/right) settings Adjust the smart exit search area parameters Increase the High Approach Boost value
7	Box collision to the mounting base	Change box rotation if possible Consider using the inverse approach Adjust the **rf_collision_radius_max** value
8	Robot collision to the mounting base	Try to pick multiple boxes at once Change the box rotation if possible Use enforced gripper rotation at pickup Consider using the inverse approach Adjust the **rf_collision_radius_max** value
9	Uneven space between adjacent boxes on the pallet	Pickup position is not properly calibrated TCP settings are not correct Box dimensions are not properly measured
10	4-ways gripper optimization is not choosing the optimal position	Move the robot closer to the pickup position Consider using dynamic positioning (in projects with lifting column)
11	Robot is repeating the same	Check that the vacuum sensor feedback is

	position over and over again	properly configured Ensure the TaskCompleted variable is properly used in the callbacks
12	Too long delay at pick position	Adjust the **rf_grip_delay** value Check the vacuum sensor feedback Disable the vacuum sensor feedback
13	Too long delay at drop position	Adjust the **rf_grip_release_delay** value Check the vacuum sensor feedback Disable the vacuum sensor feedback
14	Program stops with error "This position cannot be palletized"	Follow the instructions in the popup message Change the box order or box rotation Consider using a lifting column with higher stroke
15	Program stops with error "Unable to find a collision-free path"	Follow the instructions in the popup message Change the box order or box rotation Adjust the **rf_lift_safe_down** value Consider using a lifting column with higher stroke
16	Program stops with "Acceleration has been reduced extremely"	Make sure the gripper dimensions are entered properly under Installation Pally Gripper Move the Smart Acceleration slider one step right
17	"Communication with controller lost" error on CB3.0-series	Reduce log-level to Warning Use inverted approach in patterns Use Lock Direction in patterns Disable 4-ways gripper optimization Use custom path
18	"Runtime thread too much behind" error on E-series	Reduce log-level to Warning When using log-level Debug, set **rf_cpuidle_ticks**=20 Use inverted approach in patterns Use Lock Direction in patterns Disable 4-ways gripper optimization Use custom path
19	Program stops immediately after starting	Make sure the product selection variables are initialized properly. Make sure the operator interface is configured properly.
20	Program starts and then waits forever without palletizing	Make sure the number of available products on the conveyor corresponds to the number of product sensors installed.

21	No trace information is available in the Log window; log messages are not visible in the Log window.	Make sure the log level is set to the requested level: Warning, Message, Info, or Debug. Make sure all 3 small push buttons (i) (!) (x) in the Polyscope Log window are pressed down.
22	Changing path planning settings has no effect.	Make sure the offline path storage database is cleared.
23	Installation node or Program node has missing tabs and settings.	Make sure the *Service and Support plan* has not expired. Obtain a new license if necessary.