OptoFidelity Touch and Test

Python Client Reference

1 Getting started

TnT python client has been tested with python 3.5 in 32-bit Windows, 64-bit Windows, 32-bit MacOs, and 64-bit MacOs environments.

To get started, make sure that TnT Server is running. Then import TnTClient module and create an instance of TnTRobotClient or TnTDUTClient class.

To control robot in robot workspace frame, create an instance of TnTRobotClient class. TnTClient class implements some factory methods to help creating other robot control classes such as TnTRobotClient.

Example:

```
from tntclient.tnt_client import TnTClient
client = TnTClient()
robot = client.robot("Robot1")
robot.move(20.0, 20.0, 10.0) # example command
```

To control robots and actuators in DUT coordinate frame, create an instance of class TnTDUTClient. This can be done by first creating TnTClient object with required initialization and then by creating TnTDUTClient.

Example:

```
from tntclient.tnt_client import TnTClient
tntclient = TnTClient()
robot = tntclient.robot("Robot1")
dut = tntclient.dut("DUT1")
dut.robot = robot
dut.jump(x=20.0, y=20.0, z=10.0, jump_height=40.0) # example command
```

More examples can be found under examples/ directory in the TnT Client package.

2 Units

In function parameters, following units are assumed unless otherwise stated:
- Linear length: millimeters [mm]
- Angle: degrees [deg]
- Time: seconds [s]
- Mass: grams [g]
Such parameters are represented with floating point types so that decimal values are allowed.

3 Module: `tnt_camera_client`

3.1 Class: `TnTCameraClient`

TnT™ Compatible Camera resource Should work together with - TnT Sequencer - TnT Positioning Tool - TnT UI
Camera coordinate convention: Camera node's parent frame determines the camera mount frame. Convention is that camera mount frame local z-axis points in the direction of the camera view (towards imaging target). When looking along camera view, the local x-axis of camera mount frame points to the left meaning that the pixel x-coordinates increase along negative local x-axis direction. The local y-axis direction points down in the image meaning that pixel y-coordinates increase along positive local y-axis direction.

Method: `detect_icon`

Description:

  DEPRECATED! List of detected objects with their positions in relation to given context.

Arguments:

icon	Name of the icon. You can teach new icons with TnT UI.
confidence	Confidence factor 0..1 where 1 is very confident. Normal confidence is about 0.7.
context	Returned positions are relative to this context.
detector	Detector name as string, like "halcon".
exposure	Exposure in seconds. If not set, uses last exposure used.
gain	Gain value. If not set, uses last gain value used.

Returns:

  List of detected objects.

Method: `focus_height`

Description:

  Get camera focus height (distance)

Arguments:

  None

Returns:

  Focus height in mm.

Method: `get_parameter`

Description:

  Get parameter value.

Arguments:

name	Name of the parameter to be read.

Returns:

  {'status': 'ok', 'params': parameter_dictionary}.

Method: `get_parameters`

Description:

  Get given set of parameters from camera. Input dict indicates which parameters are retrieved.

Arguments:

parameters	Additional arguments to indicate which parameters should be retrieved from camera. Values are ignored. List cannot be passed into HTTP GET that only has arguments and no body.

Returns:

Dictionary of requested parameters and their values.

Method: `read_text`

Description:

DEPRECATED! Take a photo and analyze the image for text.

Arguments:

context	Returned positions are relative to this context.
language	Language as string, like "English" (default).
detector	Analyzer name, like abbyy (default).
exposure	Exposure in seconds. If not set, uses last exposure used.
gain	Gain value. If not set, uses last gain value used.

Returns:

Found text with position information.

Method: `take_still`

Description:

Takes a photo. This function opens camera automatically, no need to use open() -function. In case the still is taken in the middle of stream view, a sleep-command might be needed before taking still to ensure that all the images from continuous capture are processed before clearing the queue.

Arguments:

filetype	jpg, png, raw(np.array).
width	optional target image width. both width and height needed
height	optional target image height. both width and height needed
zoom	optional image zoom.
undistorted	true/false, only available after undistort called succesfully
exposure	exposure in seconds. If not set, uses last exposure used.
gain	gain value. If not set, uses last gain value used.
scaling	Camera image scaling factor in range [0, 1].
interpolation	Interpolation method ('nearest', 'linear', 'cubic').
flipx	mirror x-coordinates of the image
flipy	mirror y-coordinages of the image
rotate90	rotate the image 90 degrees
duration	Video capture duration in seconds. Per-pixel maximum will be taken over the video.

Returns:

image in 'filetype' format In case filetype is "bytes", the returned image is bytearray that can be transformed into numpy array as: data = camera_client.take_still(filetype="bytes") w = int.from_bytes(data[0:4], byteorder="big") h = int.from_bytes(data[4:8], byteorder="big") d = int.from_bytes(data[8:12], byteorder="big") data = np.frombuffer(data[12:], dtype=np.uint8).reshape((h, w, d))

Method: `close`

Description:

Shuts down the camera.

Arguments:

None

Method: `move`

Description:

Move camera focus point to given position (x, y, and z-coordinate) in a given context.

Arguments:

x	Target x coordinate in a given context.
y	Target y coordinate in a given context.
z	Target z coordinate in a given context.
context	Name of the target context.

Method: `open`

Description:

Open camera for use. After the camera has been opened, sequential images can be taken quickly.

Arguments:

None

Method: `set_parameter`

Description:

Set parameter value.

Arguments:

name	Name of the parameter to set.
value	Value to set to the parameter.

Returns:

{'status': 'ok'}

Method: `set_parameters`

Description:

Set camera parameters. See get_parameters.

Arguments:

parameters	Parameter dictionary.

Returns:

{'status': 'ok'}

Method: `start_continuous`

Description:

Start continuous camera image capture. Note that Camera node argument max_queue_size should be small such as 1 to avoid latency.

The mjpeg stream can be retrieved via GET request at URL

http://127.0.0.1:8000/tnt/workspaces/ws/cameras/Camera1/mjpeg_stream where Camera1 is the name of the camera node.

This can be used in HTML image element to view the stream. Be aware that browser can cache the image so if may be necessary to add e.g. current time as parameter to the URL by importing time library and adding '?'+str(time.time()) to the URL

Arguments:

width	Width of image or None to use full image width.
height	Height of image or None to use full image width.
zoom	Zoom factor or None for no zoom.
undistorted	Use distortion calibration to undistort?

www.optofidelity.com   sales@optofidelity.com

exposure	Camera exposure or None to use Camera node exposure state.
gain	Camera gain or None to use Camera node exposure state.
scaling	Scaling factor or None for no scaling.
interpolation	Interpolation method ("nearest", "linear" or "cubic"). May affect image stream performance.
trigger_type	Camera triggering based on internal timer "SW" or external trigger "HW".
target_context	Possible DUT target for image context transformation, "None" means no transformation. (Default: None)
target_context_margin	Margin for context transformation. (Default: 0)

Method: `stop_continuous`

Description:

  Stop continuous capture that was started by start_continuous().

Arguments:

  None

www.optofidelity.com

sales@optofidelity.com

4 Module: `tnt_dut_client`

4.1 Class: `TnTDUTClient`

TnT™ Compatible DUT resource Should work together with - TnT Sequencer - TnT Positioning Tool

Method: `info`
Description:
  Read raw info from DUT device
Arguments:
  None
Returns:
  info dictionary

Method: `list_buttons`
Description:
  List button names of the current DUT.
Arguments:
  None
Returns:
  List of button names.

Method: `region_contour`
Description:
  Return the given region as a list of (x, y) points. "contour" as in how OpenCV names the approximation of a shape as a point list: 'Contours can be explained simply as a curve joining all the continuous points (along the boundary)' Can be used to OpenCV shape analysis or any other shape-analysis or geometric operation.
Arguments:

region	Name of the region.
num_points	Number of points to use on contour.

Returns:
  List of (x, y) contour points, in millimeters. Scaling is applied.

Method: `get_robot_position`
Description:
  Get the current robot position in DUT coordinates. DEPRECATED: Added for client compatibility.
Arguments:

robot_name	Name of robot whose position to get.

Returns:
  Robot position in DUT coordinates.

Method: `svg_data`
Description:
  Return the svg file specified for the dut. If there is no specified file, return empty string
Arguments:
  None
Returns:
  svg file or empty string

Method: `touches`

Description:

Gets list of touches since last call of this function. If you want to clear the touches buffer, call this function and discard the results.

Current OptoTouch application supports the following fields: x   Touch x-coordinate. Touch resolution is the same as screen pixel resolution. y   Touch y-coordinate. Touch resolution is the same as screen pixel resolution. pressure   Touch pressure, float value in range 0..1, device dependent and not any standard unit. id   Touch id. Sequential number where every new finger to touch the screen takes the first free positive number as id. action   Numbered enumeration where: 0 = touch start 1 = touch end 2 = touch move / stays pressed at point 3 = touch cancelled orientation Stylus event; angle in radians where 0 == north, -pi/2 = west, pi/2 = east azimuth   Stylus event; angle in radians where 0 == normal to surface and M_PI/2 is flat to surface. distance   Stylus event; 0.0 indicates direct contact and larger values indicate increasing distance from the surface.

Arguments:

None

Returns:

(dict) where key 'fields' is a list of touch field names. and where key 'touches' is a list of touches. One touch is an array of values.

Method: `find_objects`

Description:

Find an object model (.shm) from the currently visible portion of the screen.

Arguments:

`filename`	Name of the icon. This is the same as the icon filename without extension.
`min_score`	Minimum accepted confidence score of result [0.0 .. 1.0] (default 0.8).
`crop_left`	Left coordinate for cropping rectangle. If None then 0 is used.
`crop_upper`	Upper coordinate for cropping rectangle. If None then 0 is used.
`crop_right`	Right coordinate for cropping rectangle. If None then maximum value is used.
`crop_lower`	Lower coordinate for cropping rectangle. If None then maximum value is used.
`crop_unit`	Unit of crop coordinates One of "per", "pix" or "mm".
`exposure`	Exposure in seconds. If not set, uses last exposure used.
`gain`	Gain value. If not set, uses last gain value used.
`detector`	detector node to be used in object recognition
`offset_x`	Take the picture with an offset in x from the DUT display center. (default is 0)
`offset_y`	Take the picture with an offset in y from the DUT display center. (default is 0)
`camera_id`	Name of the camera that is being used. (default is Camera1)
`duration`	Video capture duration in seconds. Per-pixel maximum is taken over the video frames. If set to None only one screenshot is taken.

Returns:

Dictionary with keys: "success", "screenshot", "results". Response body: success -- Always True results -- Array of Result objects screenshot -- Name of the screenshot Result object (dictionary): score -- Match score [0.0 .. 1.0] topLeftX_px -- Bounding box top left x coordinate in pixels topLeftY_px -- Bounding box top left x coordinate in pixels bottomRightX_px -- Bounding box bottom right x coordinate in pixels bottomRightY_px -- Bounding box bottom right y coordinate in pixels centerX_px -- Bounding box center x coordinate in pixels centerY_px -- Bounding box center y coordinate in pixels topLeftX -- Bounding box top left x coordinate in mm topLeftY -- Bounding box top left x coordinate in mm bottomRightX -- Bounding box bottom right x coordinate in mm bottomRightY -- Bounding box bottom right y coordinate in mm centerX -- Bounding box center x coordinate in mm centerY -- Bounding box center y coordinate in mm shape -- Name of the shape file scale -- Detected icon scale angle -- Detected icon angle

Method: `screenshot`

Description:

Take screenshot of DUT screen and save it as image. First moves robot so that camera is in the middle of the DUT.

Arguments:

www.optofidelity.com

sales@optofidelity.com

camera_id	Name of camera to take image with.
crop_left	Left coordinate for cropping rectangle. If None then 0 is used.
crop_upper	Upper coordinate for cropping rectangle. If None then 0 is used.
crop_right	Right coordinate for cropping rectangle. If None then maximum value is used.
crop_lower	Lower coordinate for cropping rectangle. If None then maximum value is used.
crop_unit	Unit of crop coordinates One of "per", "pix" or "mm".
exposure	exposure in seconds. If not set, uses last exposure used.
gain	gain value. If not set, uses last gain value used.
offset_x	Take the picture with an offset in x from the DUT display center. (default is 0)
offset_y	Take the picture with an offset in y from the DUT display center. (default is 0)

Returns:

Image name.

Method: `search_text`

Description:

Search text pattern from the visible portion of the DUT screen.

Arguments:

pattern	Text or pattern to find. Search all text with pattern "" (default).
regexp	Use pattern as a regexp. [True | False (default)].
language	OCR language e.g. "English" (default), "Finnish".
min_score	Minimum score (confidence value). 0.0 - 1.0. Default is 0.8, value over 0.6 means the sequences are close matches.
case_sensitive	Should the comparison be done case sensitive or not. [True (default) | False]
crop_left	Left coordinate for cropping rectangle. If None then 0 is used.
crop_upper	Upper coordinate for cropping rectangle. If None then 0 is used.
crop_right	Right coordinate for cropping rectangle. If None then maximum value is used.
crop_lower	Lower coordinate for cropping rectangle. If None then maximum value is used.
crop_unit	Unit of crop coordinates One of "per", "pix" or "mm".
exposure	Exposure in seconds. If not set, uses last exposure used.
gain	Gain value. If not set, uses last gain value used.
detector	Name of text detector to use.
offset_x	Take the picture with an offset in x from the DUT display center. (default is 0)
offset_y	Take the picture with an offset in y from the DUT display center. (default is 0)
filter	Name of filter to apply to the image before text search. None to use no filtering (default).
camera_id	Name of the camera that is being used. (default is Camera1)

Returns:

Dictionary with keys "success", "results" and "screenshot". Response body: success -- Always True results -- Array of Result objects screenshot -- Name of the screenshot The results array is ordered in page order i.e. text found from left to right and top to bottom. Note that this means that the first item might not have the highest score. Result object (dictionary): score -- Match score [0.0 .. 1.0] topLeftX_px -- Bounding box top left x coordinate in pixels topLeftY_px -- Bounding box top left x coordinate in pixels bottomRightX_px -- Bounding box bottom right x coordinate in pixels bottomRightY_px -- Bounding box bottom right y coordinate in pixels centerX_px -- Bounding box center x coordinate in pixels centerY_px -- Bounding box center y coordinate in pixels topLeftX -- Bounding box top left x coordinate in mm topLeftY -- Bounding box top left x coordinate in mm bottomRightX -- Bounding box bottom right x coordinate in mm bottomRightY -- Bounding box bottom right y coordinate in mm centerX -- Bounding box center x coordinate in mm centerY -- Bounding box center y coordinate in mm

Method: `filter_lines`

Description:

Filter list of (x1, y1, x2, y2) lines with given region and margin. The filter will cut lines to pieces that fit inside the region with given

www.optofidelity.com

sales@optofidelity.com

margin. One given line can result to none or several lines.

Arguments:

lines	List of (x1, y1, x2, y2) lines, in millimeters.
region	Name of the filter region.
margin	Margin inwards the region, millimeters.

Returns:

List of list of (x1, y1, x2, y2) lines (each given line results to a list of lines).

Method: filter_points

Description:

Filter a list of points that are inside of DUT shape, given region and margin.

Arguments:

points	List of (x, y) points, millimeters.
region	Name of the region.
margin	Margin inwards the given region, millimeters.

Returns:

Filtered list of (x, y) points.

Method: move

Description:

Moves in to given DUT position via straight path. DEPRECATED: Added for client compatibility.

Arguments:

x	Target x coordinate on DUT.
y	Target y coordinate on DUT.
z	Target z coordinate on DUT.
tilt	Tilt angle in DUT frame (default: 0).
azimuth	Azimuth angle in DUT frame (default: 0).
robot_name	Name of robot to move.

Method: show_image

Description:

Shows image on DUT screen. If image is bytes object, use: image = base64.decodebytes(image.encode("ascii"))

Arguments:

image	base64 encoded data. None will empty the screen

Returns:

"ok" / error

Method: circle

Description:

Performs a circle movement with given parameters.

Arguments:

x	Circle center x coordinate on DUT.
y	Circle center y coordinate on DUT.
r	Circle radius.
n	How many circles should be moved. Can be floating point.

angle	Start angle in degrees. For 0 angle, start x is x+r and start y is y.
z	Target z coordinate on DUT when hovering before and after gesture (default: DUT's base_distance).
tilt	Tilt angle in DUT frame (default: 0).
azimuth	Azimuth angle in DUT frame (default: 0).
clearance	Coordinate z on dut when moving in circle (default: 0).
clockwise	True if moving clockwise, false to move counter clockwise (default: false).
separation	Separation during circle. If None, then default separation is used.
tool_name	Name of tool to perform circle with. One of 'tool1', 'tool2', 'both' or None. None is the same as 'tool1'.

Method: `double_tap`

Description:

   Performs a double tap with given parameters.

Arguments:

x	Target x coordinate on DUT.
y	Target y coordinate on DUT.
z	Target z coordinate on DUT when hovering before and after gesture (default: DUT's base_distance).
tilt	Tilt angle in DUT frame (default: 0).
azimuth	Azimuth angle in DUT frame (default: 0).
clearance	Target z coordinate on DUT when tapping (default: 0).
duration	How long to keep finger down in seconds (default: 0s).
interval	How long to pause between taps in seconds (default: 0s).
separation	Separation during tap. If None, then default separation is used.
tool_name	Name of tool to perform tap with. One of 'tool1', 'tool2', 'both' or None. None is the same as 'tool1'.

Method: `drag`

Description:

   Performs a drag with given parameters.

Arguments:

x1	Start x coordinate on DUT.
y1	Start y coordinate on DUT.
x2	End x coordinate on DUT.
y2	End y coordinate on DUT.
z	Target z coordinate on DUT when hovering before and after gesture (default: DUT's base_distance).
tilt1	Start tilt angle in DUT frame.
tilt2	End tilt angle in DUT frame.
azimuth1	Start azimuth angle in DUT frame.
azimuth2	End azimuth angle in DUT frame.
clearance	Z coordinate on DUT when running drag (default: 0)
predelay	Delay between touchdown and move in seconds.
postdelay	Delay between the end of movement and touchup in seconds.

separation	Separation during drag. If None, then default separation is used.
tool_name	Name of tool to perform drag with. One of 'tool1', 'tool2', 'both' or None. None is the same as 'tool1'.

Method: `drag_force`

Description:

Performs a drag with force with given parameters.

Force can be applied with tool1, tool2 or with both simultaneously. Note that the tool that is used must have a valid force calibration table in the configuration. Otherwise exception is raised. In case both tools are used, the given force is per tool so that force=100 & tool_name="both" will apply 200 gF on the target.

Arguments:

x1	Start x coordinate on DUT.
y1	Start y coordinate on DUT.
x2	End x coordinate on DUT.
y2	End y coordinate on DUT.
force	Grams of force to apply when running on DUT surface.
z	Target z coordinate on DUT when hovering before and after gesture (default: DUT's base_distance).
tilt1	Start tilt angle in DUT frame.
tilt2	End tilt angle in DUT frame.
azimuth1	Start azimuth angle in DUT frame.
azimuth2	End azimuth angle in DUT frame.
separation	Separation during drag force. If None, then default separation is used.
tool_name	Name of tool to perform drag force with. One of 'tool1', 'tool2', 'both' or None. None is the same as 'tool1'.

Method: `jump`

Description:

Performs a jump with given parameters. In case jump_height is not given, robot jumps to maximum height along robot z axis.

Arguments:

x	Target x coordinate on DUT.
y	Target y coordinate on DUT.
z	Target z coordinate on DUT.
jump_height	Height of the jump from DUT surface (default: jump to robot maximum height).

Method: `multi_tap`

Description:

Performs multiple tap gestures in DUT context according to given points. Start and end of multi-tap sequence is the base distance of the DUT.

Arguments:

points	List of TnTDUTPoint objects indicating where to tap. Z coordinates of the points are ignored and lift and clearance are used for tap movement.
lift	Distance of how high the effector is raised from the DUT between the taps.
clearance	Z coordinate on DUT on tap down (default: 0).

Method: `path`

Description:

Performs path movement through given points.

Arguments:

points	List of points to go through.
clearance	Clearance added to z value of each point.
separation	Separation during path. If None, then default separation is used.
tool_name	Name of tool to perform path with. One of 'tool1', 'tool2', 'both' or None. None is the same as 'tool1'.

Method: `pinch`

Description:

Performs a pinch gesture in DUT context.

Arguments:

x	Target x coordinate on DUT. Target is the middle position between fingers.
y	Target y coordinate on DUT. Target is the middle position between fingers.
d1	Distance between fingers at the beginning.
d2	Distance between fingers at the end.
azimuth	Azimuth angle during the pinch.
z	Overrides base distance.
clearance	Distance from DUT surface during gesture.

Returns:

"ok" / error

Method: `press`

Description:

Performs a press gesture in DUT context.

Force can be applied with tool1, tool2 or with both simultaneously. Note that the tool that is used must have a valid force calibration table in the configuration. Otherwise exception is raised. In case both tools are used, the given force is per tool so that force=100 & tool_name="both" will apply 200 gF on the target.

Arguments:

x	Target x coordinate on DUT.
y	Target y coordinate on DUT.
force	Force in grams, to be activated after moving to lower position.
z	Target z coordinate on DUT when hovering before and after gesture (default: DUT's base_distance).
tilt	Tilt angle in DUT frame (default: 0).
azimuth	Azimuth angle in DUT frame (default: 0).
duration	How long to keep specified force active in seconds (default: 0s).
press_depth	Distance from DUT surface during press, negative values being below/through DUT surface. Tip will be pressed to the desired depth if the force limit is not reached before that. If the force limit is reached before desired depth, the movement will stop there. (default: -1mm).
separation	Separation during press. If None, then current separation is used.
tool_name	Name of tool to perform press with. One of 'tool1', 'tool2', 'both' or None. None is the same as 'tool1'.

Returns:

"ok" / error

Method: `swipe`

Description:

Performs a swipe with given parameters. Robot accelerates and decelerates along an arc of given radius before and after touching the DUT.

Arguments:

www.optofidelity.com   sales@optofidelity.com

x1	Swipe start x coordinate on DUT.
y1	Swipe start y coordinate on DUT.
x2	Swipe end x coordinate on DUT.
y2	Swipe end y coordinate on DUT.
tilt1	Swipe start tilt.
tilt2	Swipe end tilt.
azimuth1	Swipe start azimuth.
azimuth2	Swipe end azimuth.
clearance	(optional) distance from DUT surface during movement
radius	Swipe radius.
separation	Separation during swipe. If None, then default separation is used.
tool_name	Name of tool to perform swipe with. One of 'tool1', 'tool2', 'both' or None. None is the same as 'tool1'.

Returns:
   "ok" / error

Method: `tap`
Description:
   Performs a tap with given parameters.
Arguments:

x	Target x coordinate on DUT.
y	Target y coordinate on DUT.
z	Target z coordinate on DUT when hovering before and after gesture (default: DUT's base_distance).
tilt	Tilt angle in DUT frame (default: 0).
azimuth	Azimuth angle in DUT frame (default: 0).
clearance	(optional) distance from DUT surface during movement
duration	How long to keep finger down in seconds (default: 0s).
separation	Separation during tap. If None, then default separation is used.
tool_name	Name of tool to perform tap with. One of 'tool1', 'tool2', 'both' or None. None is the same as 'tool1'.
kinematic_name	Name of kinematic to perform tap with. One of 'tool1', 'tool2', 'mid', 'synchro' or None. None is the same as 'tool1'.

Returns:
   "ok" / error

Method: `watchdog_tap`
Description:
   Performs a tap designed for HW level triggering based on when the DUT is touched. Example use case is watchdog measurement.
Arguments:

x	Target x coordinate on DUT.
y	Target y coordinate on DUT.
z	Target z coordinate on DUT when hovering before and after gesture (default: DUT's base_distance).
tilt	Tilt angle in DUT frame (default: 0).

www.optofidelity.com

sales@optofidelity.com

`azimuth`	Azimuth angle in DUT frame (default: 0).
`clearance`	(optional) distance from DUT surface during movement
`duration`	How long to keep finger down in seconds (default: 0s).
`trigger_direction`	Trigger when touching the DUT. None means that no triggering is done. "TOUCH_START" triggers when tool goes down to start touching the DUT. "TOUCH_END" triggers when tool goes up from DUT detach from it.

Returns:

"ok" / error

Properties:

`base_distance`	DUT base distance.
`bl`	Position of bottom left corner of the DUT.
`bottom_left`	Bottom left corner. DEPRECATED: Added for client compatibility.
`bottom_right`	Bottom right corner. This property can't be set. The value is calculated from tl, tr and bl. DEPRECATED: Added for client compatibility.
`br`	Position of bottom right corner of the DUT. This property can't be set. The value is calculated from tl, tr and bl.
`height`	DUT height in mm.
`orientation`	Orientation of the DUT i.e. the basis vectors of DUT's transform matrix. Given as dictionary {'i': x_basis_vector, 'j': y_basis_vector, 'k': z_basis_vector}.
`position`	Corner positions of the DUT in its relative coordinate system context. Property is in dictionary from {"top_left": top_left, "top_right": top_right, "bottom_left": bottom_left, "bottom_right": bottom_right}. DEPRECATED: Exists for client compatibility.
`tl`	Position of top left corner of the DUT.
`top_left`	Top left corner. DEPRECATED: Added for client compatibility.
`top_right`	Top right corner. DEPRECATED: Added for client compatibility.
`tr`	Position of top right corner of the DUT.
`width`	DUT width in mm.

5 Module: `tnt_robot_client`

5.1 Class: `TnTRobotClient`

Method: `get_active_finger`
Description:
  Get active finger.
Arguments:
  None
Returns:
  Active finger ID.

Method: `get_attached_tips`
Description:
  Get names of tips attached to the robot.
Arguments:
  None
Returns:
  Dictionary where key is tool name and value is name of tip attached to that tool. If no tip is attached then value is None.

Method: `get_finger_separation`
Description:
  Get separation of two fingers in mm.
Arguments:
  None
Returns:
  Separation distance in mm.

Method: `finger_separation_limits`
Description:
  Get minimum and maximum axis-to-axis separation values.
Arguments:
  None
Returns:
  Minimum and maximum values as list [min_separation, max_separation].

Method: `get_position`
Description:
  Returns the current robot position in given context.
Arguments:

`context`	Name of context where to evaluate the position.

www.optofidelity.com   sales@optofidelity.com

details	Return detailed position info.
kinematic_name	Name of kinematic that corresponds to the position.

Returns:

  Dictionary with keys 'position' and 'status'.

Method: get_speed

Description:

  Returns robot's current speed and acceleration.

Arguments:

  None

Returns:

  Dictionary with keys 'speed' and 'acceleration'.

Method: set_active_finger

Description:

  Set active finger. Kinematics are applied to the active finger so that it is possible to command either of the two fingers to specified pose.

Arguments:

finger_id	ID of finger to set active. 0=axial finger, 1=separated finger.

Method: change_tip

Description:

  Commands robot to change new tool tip from tip holder. DEPRECATED: This is 2-finger TPPT compatibility and does the same as put_attach_tip().

Arguments:

tip	Name of tip to make the current tip.
finger_id	ID of finger where to change tip to. Ignored if robot has only one tool.
attach_manually	If tip is to be attached manually and not with the tip changer.

Returns:

  Name of the tip that was attached as dict {"tip": tip_name}

Method: detach_tip

Description:

  Detach tip from robot finger if one is attached.

Arguments:

tool_name	Name of tool node where tip is detached from. Must be a child of Mount node.
finger_id	ID of finger.
detach_manually	If tip is to be detached manually and not with the tip changer.

Returns:

  Name of the tip that was detached as dict {"tip": tip_name}.

Method: set_finger_separation

Description:

  Set separation of two fingers in mm. Separation distance is measured from finger axes.

Arguments:

distance	Distance in mm.
kinematic_name	Name of kinematic to use for the motion. If None then currently active kinematic is used.

Returns:

  Dictionary with "status" key.

Method: `go_home`

Description:

  Commands the robot to go into home position.

Arguments:

  None

Method: `move`

Description:

  Moves robot into a given location using a linear motion. Coordinates and angles are interpreted in given context. Values tilt and azimuth are taken from Euler angles for global static y and z axes (in selected context) that are applied in order y, z. Tilt is angle around y-axis and azimuth is negative angle around z-axis.

Arguments:

x	Target x coordinate.
y	Target y coordinate.
z	Target z coordinate.
tilt	Euler angle for static y-axis (default: 0) in selected context.
azimuth	Negative Euler angle for static z-axis (default: 0) in selected context.
context	Name of context where coordinates and angles are interpreted.

Method: `move_relative`

Description:

  Moves robot axes by specified distance using linear motion. Parameters are relative to current position and can hence be negative or positive.

Arguments:

x	Relative x axis movement.
y	Relative y axis movement.
z	Relative z axis movement.
tilt	Relative tilt axis movement.
azimuth	Relative azimuth axis movement.

Method: `press_physical_button`

Description:

  Performs a press gesture on the given button.

Arguments:

button_name	The button to press.
duration	How long to keep button pressed in seconds (default: 0s).

Returns:

  "ok" / error

Method: `reset_robot_error`

Description:

  Resets robot error state.

Arguments:

  None

Method: `set_speed`

Description:

  Set robot speed and acceleration.

Arguments:

speed	Linear movement speed in mm/s.
acceleration	Robot acceleration in mm/s^2.
override	Not used. DEPRECATED.

Method: `set_speed`

www.optofidelity.com

sales@optofidelity.com

6 Module: `tnt_tip_client`

6.1 Class: `TnTTipClient`

TnT™ Compatible tip resource Tip can be parented to a Tool node.

Properties:

`diameter`	Diameter of the tip.
`first_finger_offset`	Distance from the middle of multifinger tip to the first finger.
`is_multifinger`	Is tip multifinger.
`length`	Length of the tip.
`model`	Tip model ("Standard", "Multifinger").
`num_tips`	Number of tips in multifinger.
`separation`	Separation of two-finger tool required to pick multifinger.
`slot_in`	Tip's slot-in position in workspace context. In this position tip is fixed to a rack.
`slot_out`	Tip's slot-out position in workspace context. In this position tip is free from a rack. When robot attaches or detaches a tip, it will first move over slot-out position.
`tip_distance`	Axial distance of adjacent tips in multifinger.

www.optofidelity.com   sales@optofidelity.com