RIEGL-VZ ROS 2 Driver API

1. Coordinate Systems

SOCS (Scanner's Own Coordinate System):

Angle data and range data are the base for calculation of the data in the Scanner's Own Coordinate System (SOCS).

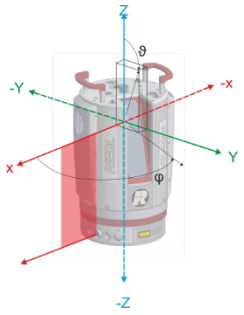

Figure 1: SOCS (Scanner's Own Coordinate System)

PRCS (Project Coordinate System):

A number of scan positions and the data acquired therein make up a scan project. The center of the project's coordinate system (PRCS) coincides horizontally with the center of the first scan position. The axes of PRCS are strictly pointing to east (x-axis, red), north (y-axis, green) and up (z-axis, blue), respectively.

The SOP transforms SOCS into PRCS (Project Coordinate System).

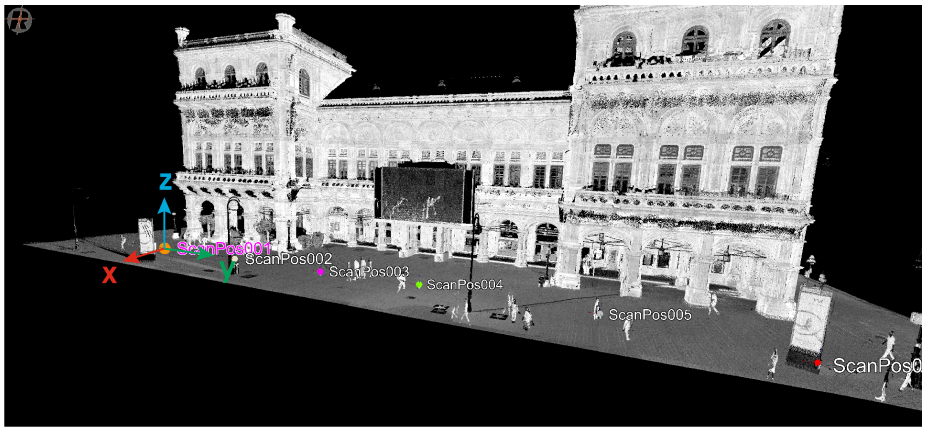

Figure 2: PRCS (Project Coordinate System)

VOCS (Voxel Coordinate System):

Automatic registration does not estimate the SOP with every new scan position, but the SOPV pose, which does not transform to PRCS, but to another cartesian coordinate system, the so called VOCS (Voxel Coordinate System). A once determined SOPV pose stays unchanged. What changes is the VOP. The VOP pose is determined via compensation of a fixed block of registered scan positions against all further measurements. Further measurements are the scanners inclination, northing from internal magnitude sensor, which is fraught with great uncertainty, and GNSS position if available.

After first scan: VOP = eye(4)
After each consecutive scan: VOP <> eye(4)

If the user is only interested in relative registration of scan positions to each other, the VOP can be ignored.

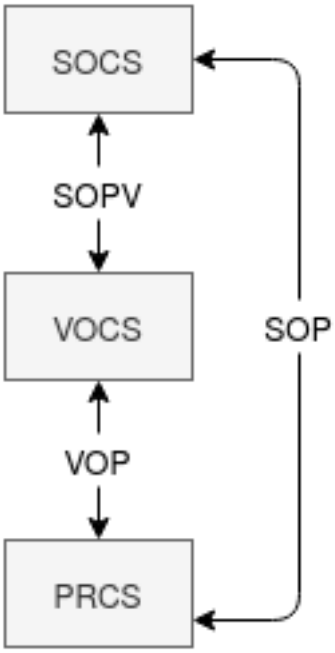

Figure 3: RIEGL Coordinate Systems

2. RIEGL Interfaces

2.1 Services

riegl_vz_interfaces/GetPointCloud:

```
uint32 index   # The scan position number within a project
---
```

```
PointCloud2 pointcloud
bool success   # indicate successful run of service
string message # informational, e.g. for error messages
```

See PointCloud2 definition: sensor_msgs/PointCloud2
A negative index implicitly refers to the last scan position, 0 is the first scan position.
The 'frame_id' in the header is either 'RIEGL_SOCS'.

riegl_vz_interfaces/GetPoses:

```
---
PoseStamped[] poses
bool success   # indicate successful run of service
string message # informational, e.g. for error messages
```

See PoseStamped definition: sensor_msgs/PoseStamped
The 'frame_id' in the header is either 'RIEGL_PRCS' or 'RIEGL_VOCS'.

riegl_vz_interfaces/SetPose:

```
PoseStamped pose
---
bool success   # indicate successful run of service
string message # informational, e.g. for error messages
```

See PoseStamped definition: sensor_msgs/PoseStamped
The 'frame_id' in the header has to be either 'RIEGL_PRCS' or 'RIEGL_VOCS'.

3. Nodes

3.1 riegl_vz

3.1.1 Parameters

~**hostname** (string, default: "") :

The scanners hostname or IP address.

~**working_dir** (string, default: "~/.ros_riegl_vz") :

The root working directory for runtime execution.

~**ssh_user** (string, default: "user") :

The linux user name for SSH login on the scanner.

~**ssh_password** (string, default: "user") :

The linux user password for SSH login on the scanner.

~**project_name** (string, default: "") :

The scan project name used by service 'set_project'. An existing project will be loaded, otherwise a new project will be created. If string is empty, a default project name will be composed from current local time and date.

~**stor_media** (integer, default: 2) :

The active storage media for scan data (1: AUTO, 2: INTERNAL SSD, 3: USB).

~**scan_pattern** (double[], default: {30.0,130.0,0.04,0.0,360.0,0.04})

Specifies the field of view (FOV) for scanning and the scan increments.
[0]: Line Start Angle
[1]: Line Stop Angle
[2]: Line Angle Increment
[3]: Frame Start Angle
[4]: Frame Stop Angle
[5]: Frame Angle Increment

~**meas_program** (integer, default: 3) :

This is the laser scanner measurement program, which specifies the laser scanner frequency.

~**scan_publish** (bool, default: "True") :

Enable publishing of point cloud data on topic 'pointcloud' after scan acquisition has finished.

~**scan_publish_filter** (string, default: "") :

Filter string for published point cloud data, e.g. "(riegl.xyz[2] > 5) && (riegl.reflectance > 35)"

~**scan_publish_lod** (integer, default: 0) :

Level of detail (LOD) for published point cloud. This is to reduce the number of measurements.
lod=0 : no reduction
lod=1 : reduce measurements by factor 2 (2^1)
lod=2 : reduce point cloud by factor 4 (2^2)
lod=3 : reduce point cloud by factor 8 (2^3)
...

~**scan_register** (bool, default: "True") :

Enable automatic scan position registration in current project after scan data acquisition has finished.

3.1.2 Published Topics

pointcloud (sensor_msgs/PointCloud2) :

Point cloud with scan data from the laser scanner in SOCS.

status (diagnostic_msgs/DiagnosticStatus.msg):

Riegl VZ status information, published once per second:

```
errors       : scanner errors ("no", "yes", "fatal")
op_state     : operating state ("ready", "scan busy", "busy")
progress     : progress of scan data acquisition or registration in percent
memory_usage : memory usage of active storage media in percent
```

3.1.3 Services

set_project (std_srvs/Trigger) :

Create a new or load an existing project on the scanner with name from parameter '~project_name'.

Response:
success = True -> message: Project Name
success = False -> message: Error Message

scan (std_srvs/Trigger) :

Start laser scan acquisition, and scan registration if parameter '~scan_register' is enabled. If parameter '~scan_publish' is enabled and laser scan has finished, scan data will be published on 'pointcloud' topic. Use 'is_scan_busy' and/or 'is_busy' services to poll background tasks busy state or get it from 'status' topic.

Response:
success = True -> message: success
success = False -> message: Error Message

is_scan_busy (std_srvs/SetBool) :

Check if scan data acquisition has finished, otherwise the device is locked. If 'data' in request is true, the call will block until background task has finished.

Request:
data: set blocking execution
Response:
success = True -> message: busy
success = False -> message: ready

is_busy (std_srvs/SetBool) :

Check if background tasks (scan data acquisition AND scan registration) have finished, otherwise the device is locked. If 'data' in request is true, the call will block until all background tasks have finished.

Request:
data: enable blocking execution mode Response:

success = True -> message: busy
success = False -> message: ready

get_pointcloud (riegl_vz_interfaces/GetPointCloud) :

Get point cloud of a previously acquired scan in actual project.

get_sopv (riegl_vz_interfaces/GetPoses) :

Request position and orientation (SOPV) of the last acquired scan.

get_all_sopv (riegl_vz_interfaces/GetPoses) :

Request positions and orientations (SOPV) of all previously acquired scans in current project.

get_vop (riegl_vz_interfaces/GetPoses) :

Get current VOP, which is the position and orientation of the voxel coordinate system (VOCS) origin based on the project coordinate system (PRCS).

set_pose (riegl_vz_interfaces/SetPose) :

Set position of the scanner origin in a referenced coordinate system (VOCS or PRCS). This is used for scan registration.

stop (std_srvs/Trigger) :

Stop laser scan data acquisition and registration background tasks.

Response:
success = True -> message: "RIEGL VZ has been stopped"

shutdown (std_srvs/Trigger) :

Stop data acquisition and power down the laser scanner.

Response:
success = True -> message: "RIEGL VZ is shutting down"

3.1.4 Extension

Not available in first implementation but for further extension:

- Providing covariance of pose (see sensor_msgs/PoseWithCovarianceStamped)
- Additional parameters:

~capture_images (bool, default: False) :

Enable capturing of camera images.

- Additional services:

get_voxel (riegl_vz_interfaces/GetPointcloud) :

Get voxel data of a previous scan data acquisition.

get_image (riegl_vz_interfaces/GetImage) :

Get camera image for scan position.

get_projectmap (riegl_vz_interfaces/GetImage) :

Get the project map overview image.