ROS2 RIEGL-VZ Package 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).

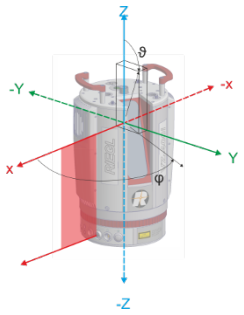

Figure 1: SOCS (Scanner's Own Coordinate System)

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.

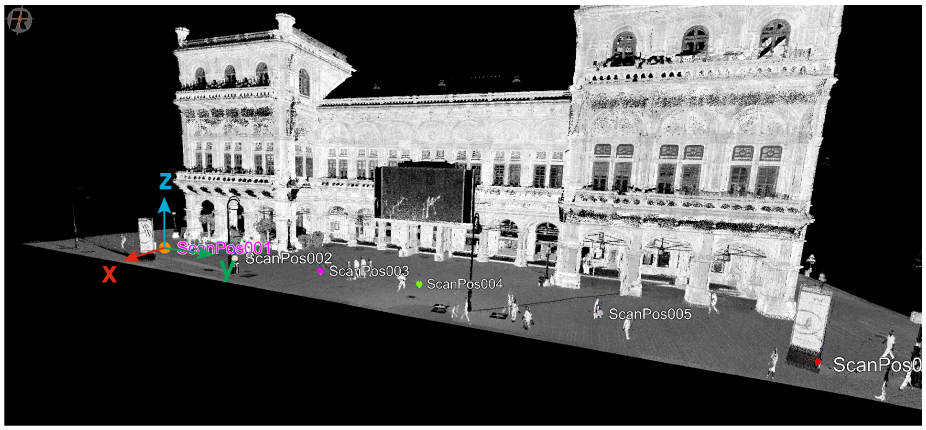

Figure 2: PRCS (Project Coordinate System)

GLCS (Global Coordinate System):

A global coordinate system like WGS84.

The POP transforms GLCS into PRCS

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 and the POP can be ignored.

2. RIEGL Interfaces

2.1 Messages

riegl_vz_interfaces/ScanPose:

```
uint32 seq   # Scan position number within a project
geometry_msgs/PoseStamped pose
```

'seq' is the scan position number.
See PoseStamped definition: geometry_msgs/PoseStamped

2.2 Services

riegl_vz_interfaces/GetPointCloud:

```
uint32 seq     # Scan position number within a project, starting with 1, 0 refers to last sc
---
sensor_msgs/PointCloud2 pointcloud
bool success   # indicate successful run of service
string message # informational, e.g. for error messages
```

See PointCloud2 definition: sensor_msgs/PointCloud2
'seq' is the scan position number, 0 implicitly refers to the last scan position, 1 is the first scan position.
The 'frame_id' in the header is 'riegl_vz_socs'.

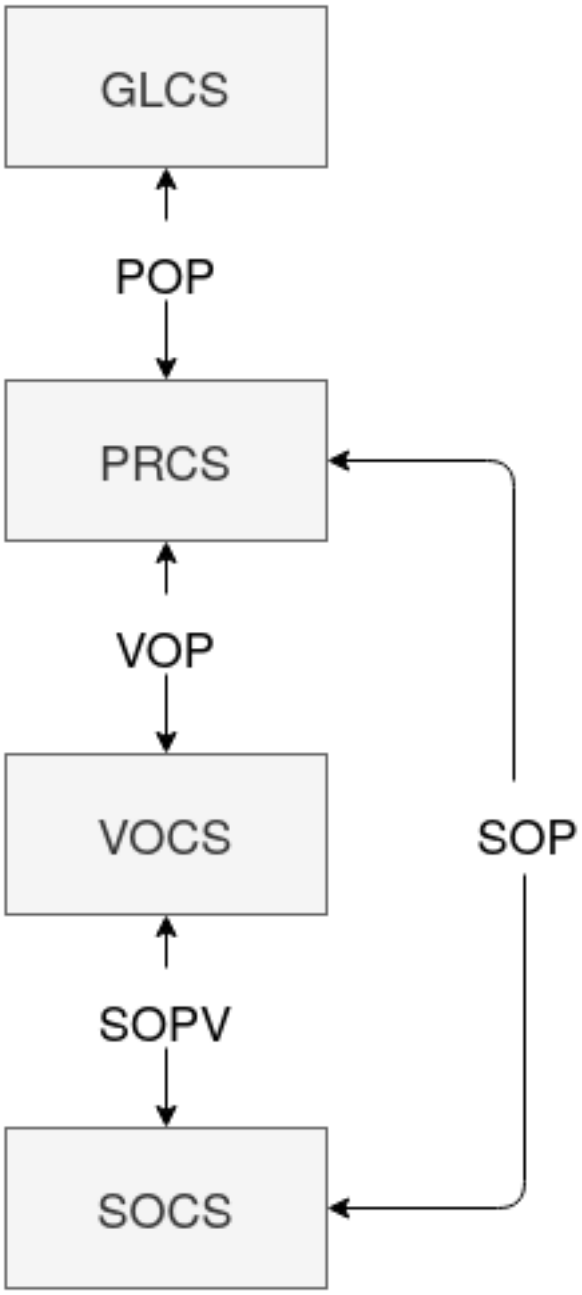

Figure 3: RIEGL Coordinate Systems

riegl_vz_interfaces/GetScanPoses:

```
---
string project                # Scan project name
ScanPose[] scanposes
geometry_msgs/PoseStamped vop # position and orientation of VOCS in PRCS
geometry_msgs/PoseStamped pop # position and orientation of PRCS in GLCS
bool success                  # indicate successful run of service
string message                # informational, e.g. for error messages
```

The 'frame_id' in the scanposes[n].header is 'riegl_vz_vocs'.
The 'frame_id' in the vop.header is 'riegl_vz_prcs'.
The 'frame_id' in the pop.header is 'riegl_vz_glcs', which is e.g. EPSG::4978

riegl_vz_interfaces/GetPose:

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

See PoseStamped definition: geometry_msgs/PoseStamped
The 'frame_id' in the pose.header is either 'riegl_vz_prcs' or 'riegl_vz_vocs'.

riegl_vz_interfaces/SetPosition:

```
geometry_msgs/PointStamped position
---
bool success   # indicate successful run of service
string message # informational, e.g. for error messages
```

See PointStamped definition: geometry_msgs/PoseStamped The 'frame_id' in the position.header is 'riegl_vz_glcs', which is e.g. EPSG::4978

3. Nodes

3.1 riegl_vz

3.1.1 Parameters

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

The scanners hostname or IP address.

~**working_dir** (string, default: "/tmp/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 name of the project to be loaded or created.

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

The active storage media for scan data recording (0: INTERNAL SSD, 1: USB, 2: SD CARD).

~**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 (2ˆ1)
lod=2 : reduce point cloud by factor 4 (2ˆ2)
lod=3 : reduce point cloud by factor 8 (2ˆ3)
...

~**scan_register** (bool, default: “True”) :

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

~**reflector_search** (bool, default: “False”) :

Enable automatic reflector search with every scan data acquisition.

~reflector_search_models (string, default: “”) :

Name of reflector search model. Can be specified multiple times, seperated by comma (e.g. “RIEGL flat reflector 50 mm, RIEGL flat reflector 100 mm”)

~reflector_search_limits (double[], default: {0.0,10000.0}) :

Minimum and maximum range in meter between scan-position and reflector.

~control_points_csv_file (string, default: “”) :

Path to CSV with control points in GLCS (Global Coordinate System).

~control_points_coord_system (string, default: “”) :

The coordinate system for control points (e.g. EPSG::4978).

3.1.2 Published Topics

pointcloud (sensor_msgs/PointCloud2) :

Point cloud with scan data from the laser scanner. Included are xyz cartesian coordinates in SOCS and reflectance in dB. Data will be published only if parameter ‘~scan_publish’ is enabled.

pose (geometry_msgs/PoseStamped):

Topic provides SOPV (Scan Position and Orientation in VOCS) of the currently registered scan position.

gnss/fix (sensor_msgs/NavSatFix.msg) :

Actual GNSS fix with position in WGS 84 coordinates, published once per second.

gnss/fix/scan (sensor_msgs/NavSatFix.msg) :

GNSS fix with position in WGS 84 coordinates, published shortly before scan data acquisition.

diagnostics (diagnostic_msgs/DiagnosticArray.msg):

Riegl VZ status information, published once per second:

```
scanner:
  opstate       : operating state ("unavailable", "waiting", "scanning", "processing")
  active_task   : active task description
  progress      : scan progress in percent
  scan_position : number of current scan position
memory:
  mem_total_gb  : total storage media memory space in GByte
  mem_free_gb   : free storage media memory space in GByte
```

```
  mem_usage     : storage media memory usage in percent of total space
gnss:
  fix           : GNSS fix
  num_sat       : number of available satellites
```

3.1.3 Services

set_project (std_srvs/Trigger) :

Load an existing project on the scanner with name from parameter '~project_name'. If the project name is empty or the project can not be loaded, a new project will be created automatically.

Response:
success = True -> message: Project Name

set_position (riegl_vz_interfaces/SetPosition) :

Set position of the scanner origin in a global coordinate system (GLCS) supported by RIEGL GeoSys Manager. The position must be set before the scan. This allows transformation of scans into a global coordinate system and furthermore the position is an initial guess for scan registration.

scan (std_srvs/Trigger) :

Start a background task for laser scan data acquisition

The execution state will be published in 'opstate' field of 'diagnostics' topic.
The node is locked until all background tasks have finished and the operating state is 'waiting' again.

If parameter '~scan_publish' is enabled, acquired data will be published on 'pointcloud' topic soon after scanning has finished.

The parameter '~scan_register' enables automatic scan position registration after scanning. The registration result is published on topic 'pose' or it can be requested by separate service calls (see 'get_sopv', 'get_vop', 'get_pop' and 'get_scan_poses') after processing, if operating state is 'waiting' again.

Response:
success = True -> message: "success"
success = False -> message: "scanner not available" | "command execution error"

get_sopv (riegl_vz_interfaces/GetPose) :

Request a single SOPV of the previously registered scan position in actual project.

Response:
success = True -> message: "success", pose: Last SOPV Pose

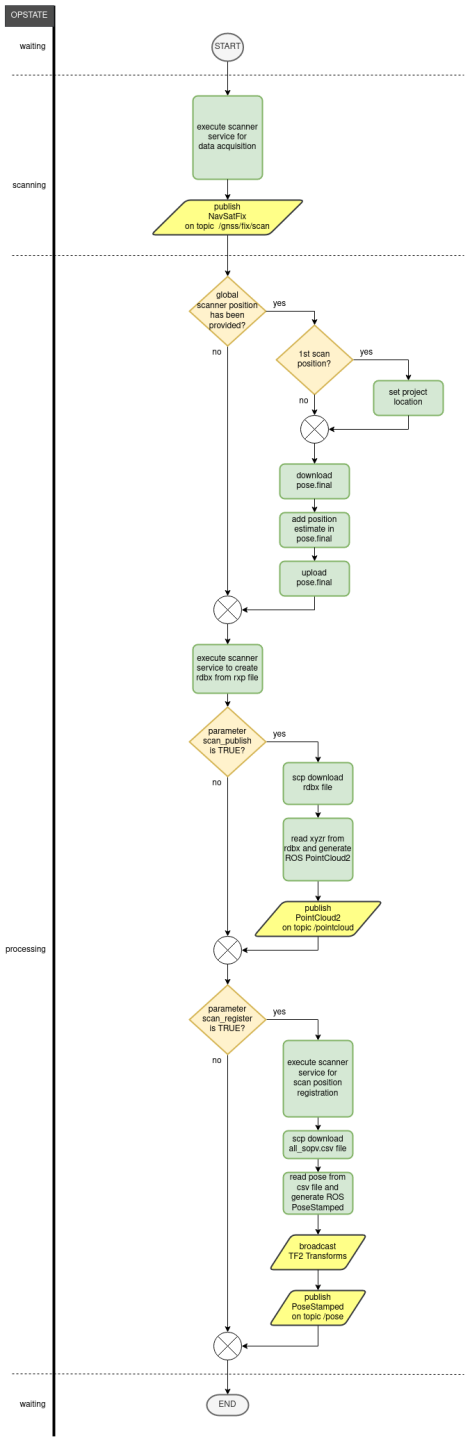

Figure 4: ROS Scan Service

success = False -> message: “scanner not available” | “command execution error”

get_vop (riegl_vz_interfaces/GetPose) :

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

Response:
success = True -> message: “success”, pose: VOP Pose
success = False -> message: “scanner not available” | “command execution error”

get_pop (riegl_vz_interfaces/GetPose) :

Get current POP, which is a single position and orientation of the project coordinate system (PRCS) origin based on the global coordinate system (GLCS).

Response:
success = True -> message: “success”, pose: POP Pose
success = False -> message: “scanner not available” | “command execution error”

get_scan_poses (riegl_vz_interfaces/GetScanPoses) :

Request all SOPVs of previously registered scan positions in actual project.

Response:
success = True -> message: “success”, project: Project Name, scanposes: All Scan Poses, vop: VOP Pose, pop: POP Pose
success = False -> message: “scanner not available” | “command execution error”

get_pointcloud (riegl_vz_interfaces/GetPointCloud) :

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

Response:
success = True -> message: “success”, pointcloud: Scan Data
success = False -> message: “scanner not available” | “command execution error”

stop (std_srvs/Trigger) :

Stop laser scan data acquisition and registration background tasks.

Response:
success = True -> message: “success”
success = False -> message: “scanner not available” | “command execution error”

shutdown (std_srvs/Trigger) :

Stop data acquisition and power down the laser scanner device.

Response:
success = True -> message: “success”
success = False -> message: “command execution error”

3.1.4 TF2 Transformation

The node will broadcast TF2 transformation messages if an existing project is loaded and after each scan position registration:

3.1.5 Extensions

Not available in first implementation but for further extension:

- Providing covariance of pose (see sensor_msgs/PoseWithCovarianceStamped)
- More diagnostic status information, e.g. scanner errors, ...
- Additional parameters:

~**capture_images** (bool, default: False) : Enable capturing of camera images.

- Additional services:

get_image (riegl_vz_interfaces/GetImage) : Get camera image for scan position.

get_voxel (riegl_vz_interfaces/GetPointcloud) : Get voxel data of a previous scan data acquisition.

get_projectmap (riegl_vz_interfaces/GetImage) : Get the project map overview image.

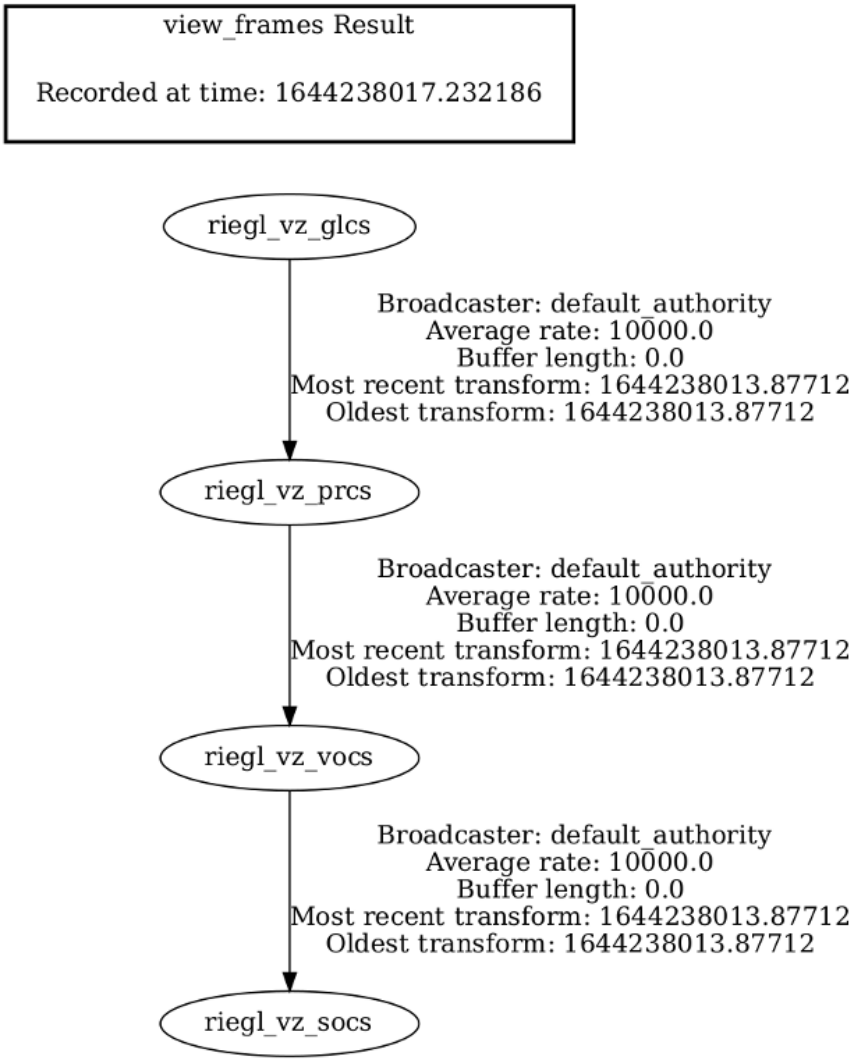

Figure 5: TF2 Transformation