ROS 2 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.

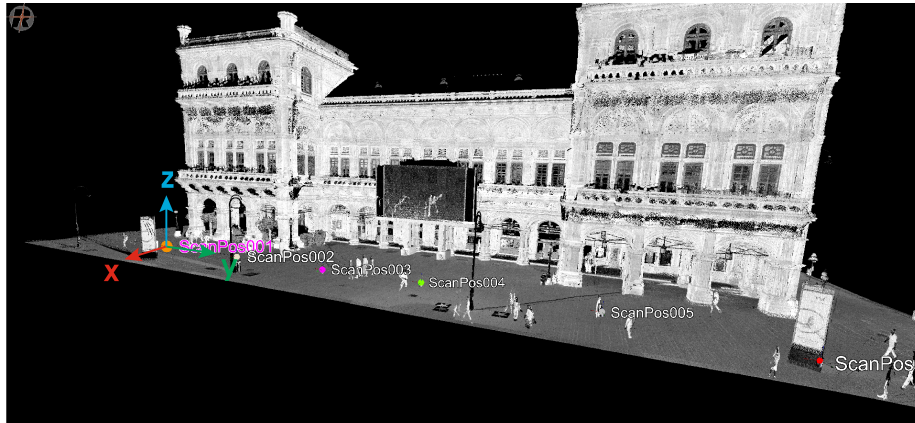The SOP transforms SOCS into PRCS (Project Coordinate System).

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: $VOP <> eye(4)$

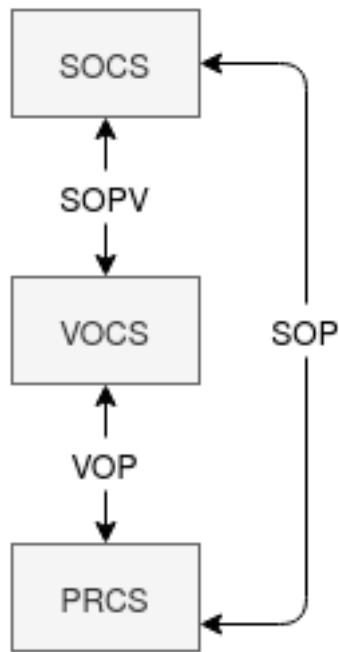If the user is only interested in relative registration of scan positions to each other, the VOP can be ignored.

Figure 3: RIEGL Coordinate Systems

2. RIEGL Interfaces

2.1 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'.

riegl_vz_interfaces/GetScanPoses:

```
---
string project                 # Scan project name
ScanPose[] scanposes
geometry_msgs/PoseStamped vop # position and orientation of VOCS in PRCS
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'.

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/SetPose:

```
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 has to be either 'riegl_vz_prcs' or
'riegl_vz_vocs'.

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.

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

The active storage media for scan data recording (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. 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.

diagnostics (diagnostic_msgs/DiagnosticArray.msg):

Riegl VZ status information, published once per second:

```
opstate      : operating state ("waiting", "scanning", "processing")
progress     : scan progress 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

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_all_sopv' and 'get_vop') after processing, if operating state is 'waiting' again.

Response:
success = True -> message: "success"
success = False -> message: "node is locked"

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: "data unavailable"

get_sopv (riegl_vz_interfaces/GetPose) :

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

Response:
success = True -> message: "success", scanposes[0]: Last SOPV Pose
success = False -> message: "data unavailable"

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: "data unavailable"

get_scan_poses (riegl_vz_interfaces/GetScanPoses) :

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

Response:
success = True -> message: "success", scanposes[..]: All SOPV Poses
success = False -> message: "data unavailable"

stop (std_srvs/Trigger) :

Stop laser scan data acquisition and registration background tasks.

Response:
success = True -> message: "success"

shutdown (std_srvs/Trigger) :

Stop data acquisition and power down the laser scanner device.

Response:
success = True -> message: "success"

OPSTATE

waiting

START

scanning

execute scanner
service for
data acquisition

execute scanner
service to create
rdbx from rxp file

Parameter
scan_publish
is TRUE?

scp download
rdbx file

read xyzr from
rdbx and generate
ROS PointCloud2

publish
PointCloud2
on topic /pointcloud

processing

Parameter
scan_register
is TRUE?

execute scanner
service for
scan position
registration

scp download
all_sopv.csv file

read pose from
csv file and
generate ROS
PoseStamped
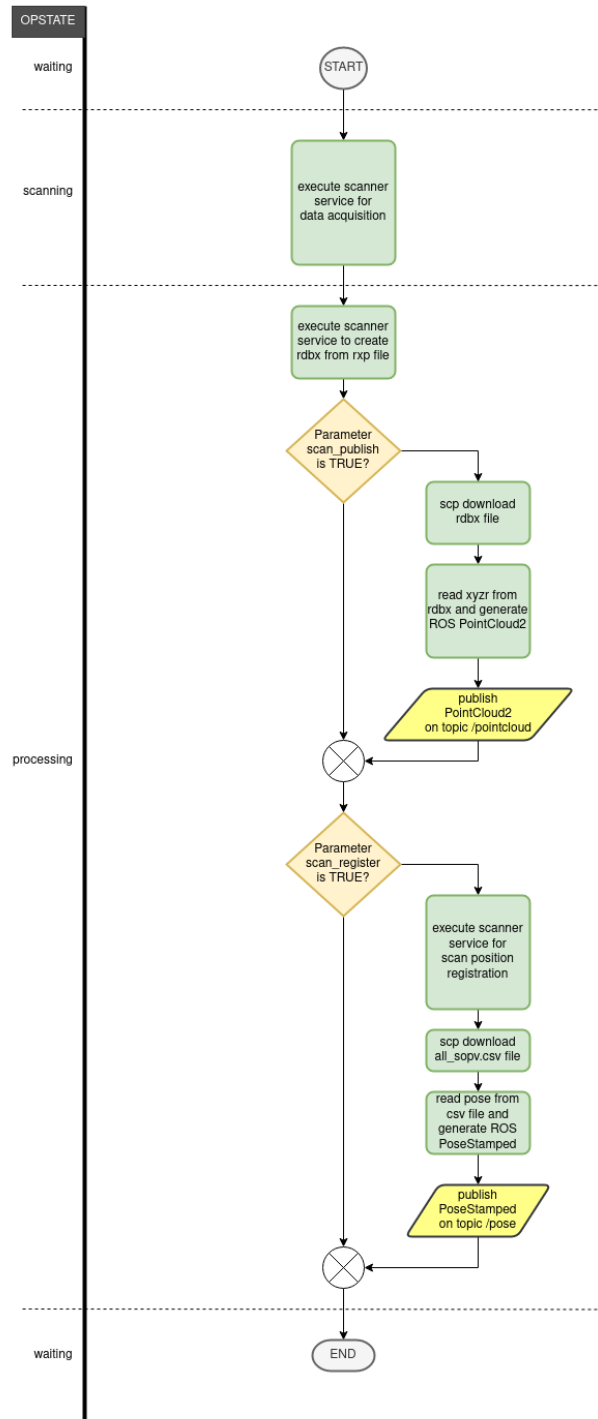
publish
PoseStamped
on topic /pose

waiting

END

Figure 4: ROS Scan Service

3.1.4 Extensions

Not available in first implementation but for further extension:

- Providing covariance of pose (see sensor_msgs/PoseWithCovarianceStamped)

- More diagnostic status information, e.g. memory usage, scanner errors

- Additional parameters:

~**capture_images** (bool, default: False) :

Enable capturing of camera images.

- Additional services:

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.

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.