Vision

# Short description

Roboteam vision is the package where the messages from the SSL organisation are received, parsed and send on as ros messages.

Additionally it keeps track of which side of the field is ours, and which color our robots are.

This is also the node that normalizes the field when requested.

# Executables

* roboteam\_vision

# Dependencies

* No direct external dependencies

# Globals

* bool us\_is\_yellow  
  Keeps track of which color our team is.  
  This allows us to put the correct robot detections into `us` and `them`.
* bool use\_legacy\_packets

Whether the incoming geometry packets are legacy or not.

This allows for replaying old SSL logs that are using the legacy format.

# Params

* our\_color  
  Either `yellow` or `blue`.  
  Initialized to `yellow` if the parameter is invalid or missing.
* our\_side  
  Either `left` or `right`.

Initialized to `right` if the parameter is invalid or missing.

Keeps track of which side of the field is ours.  
 Used when normalizing the field to determine whether rotating is necessary.

* use\_legacy\_packets

Boolean parameter, either `true` or `false`.

Initialized to `false` if the parameter is invalid or missing.

Specifies whether the incoming geometry packets are legacy or not.

This allows for replaying old SSL logs that are using the legacy format.

* normalize\_field

Boolean parameter, either `true` or `false`.

Defaults to `false` if parameter not found.

Specifies whether the field is to be normalized or not.

## Transformation params

* transform\_field/enabled  
  Boolean parameter  
  When set to `true` the field will be transformed according to the following `transform\_field` parameters. When `false` these parameters will be ignored.
* transform\_field/rotate  
  Boolean parameter  
  When `true` the detection packets sent out will be rotated 90 degrees counterclockwise.
* transform\_field/offset/[top, left, bottom, right]  
  Double parameters  
  These four parameters indicate the padding between the transformed and the real field. So a value of `3` for `top` will mean the transformed field will occupy the lower half of the real field.

# Input topics

This node has no input ropics.

# Output topics

* vision\_detection

Output for robot and ball detections.

* vision\_geometry

Output for field geometry information.

* vision\_refbox

Output for referee events.

# Details

The vision node listens for SSL multicast messages on two ip addresses:

* 224.5.23.2:10006

For `DetectionFrame` and `GeometryData` messages.

* 224.5.23.1:10003

For `RefboxCmd` messages.

Received ProtoBuf messages map directly to output topics in the following manner:

- DetectionFrame -> vision\_detection

- GeometryData -> vision\_geometry

- RefboxCmd -> vision\_refbox

Where `GeometryData` can either be the 2014 legacy version or the new version. Whether the node expects legacy packets or not depends on the `use\_legacy\_packets` parameter.

The source files of the relevant ProtoBuf messages are stored in the `roboteam\_utils` package.

## Normalization

The rest of the software nodes expect our side of the field to have a negative x coordinate (the “left” side), and the opponents side to have a positive x coordinate.

This is done because it simplifies many calculations and removes the need to check for `our\_side` before doing anything relying on who’s side is where.

The SSL vision program does not follow this convention however. Therefore this node needs to normalize the field when our side is the “right” side. The parameter that determines which side is ours is `our\_side`. The parameter for switching normalization on or off is `normalize\_field`. It is on by default.

To normalize a detection or referee packet, all coordinates are mirrored over the y axis.

Geometry packets are not normalized, as the field is symmetrical in both the x- and y axis.

De-normalization at the robothub is not needed, as the robot commands are sent out in robot-relative space, and not world-relative.

## Transformations

During tests in Nagoya, it may become necessary to play on only a portion of the field. We know this from a team that participated in 2016 and ran into the problem that they couldn’t do that.

The vision node has the capability to transform the incoming detection and geometry packets so that to the AI the field appears smaller, in a different location, or rotated a 90 degrees.

Moving and rotating is applied to the detection and referee packets only. Scaling is applied to only the geometry packets. This is because the AI expects the field to be horizontal and (0, 0) to be in the center. By only applying the scaling to the field itself and transform the rest around it, we adhere to this expectation.

The location, scale and rotation can be changed by the `transform\_field` parameters listed in the parameter section.