drySLAM (Anthony Rodriguez, Shuai Yuan, Shreenivaas Devarajan

Requirements Verification
The requirements have changed in scope: In general, larger UI tasks are less important than Matlab-approximating algorithmic tasks, which means that plotting data and integrating new sensors are no longer part of the requirements list. These requirements have been replaced by requirements regarding the successful approximation of the Matlab code.

Overview:

Requirement 1: Program adds and deletes unique corner/edge features in every iteration.
Requirement 1.1: Delete appropriate features in every iteration
Validation 1.1: Test 1.1
Test 1.1: Run *deleteFeatures* process on program pre-state exported from corresponding Matlab process and compare post-state from Matlab output file to post-state of program.

Requirement 1.2: Update appropriate features in every iteration
Validation 1.2: Test 1.2
Test 1.2: Run *updateFeatures* process on program pre-state exported from corresponding Matlab process and compare post-state from Matlab output file to post-state of program.

Requirement 1.3: Convert features at finite distances from "inverse-depth" corrdinates to Euclidian as applicable in every iteration
Validation 1.3: Test 1.3
Test 1.3: Run *inversedepth_2_cartesian* process on program pre-state exported from corresponding Matlab process and compare post-state from Matlab output file to post-state of program.

Requirement 1.4: Pgm can detect new features
Validation 1.4: Test 1.4
Test 1.4: Run *initialize_features* process and verify by inspection that (1)new features are added and (2)any new features correspond to edge/corner artifacts in image.

Requirement 2: Extended Kalman Filter predicts state and assembles covariance matrix
Requirement 2.1: Values for predicted state and covariance matrices are within 10^{-6} of Matlab impl.
Validation 2.1: Test 2.1
Test 2.1: Run *ekf_prediction* process on program pre-state exported from corresponding Matlab process and compare post-state from Matlab output file to post-state of program.

Requirement 3: Pgm can match predicted feature position data to features being actively tracked
Requirement 3.1: Camera movements are predicted within 10^{-6} of Matlab values
Validation 3.1: Test 3.1
Test 3.1: Run *predict_camera_measurements* process on program pre-state exported from corresponding Matlab process and compare post-state from Matlab output file to post-state of program.

Requirement 3.2: In each iteration, predictions are made about feature location in next iteration
Validation 3.2: Test 3.2
Test 3.2: Run *predict_features_appearance* process on program pre-state exported from corresponding Matlab process and compare post-state from Matlab output file to post-state of program.

Requirement 3.3: Predicted feature locations from (3.2) are matched to actual features in current image
Validation 3.3: Test 3.3
Test 3.3: Run *matching* process on program pre-state exported from corresponding Matlab process and compare post-state from Matlab output file to post-state of program.

Requirement 4: 1-pt RANSAC is used to select localization hypothesis based on data from Reqs (1-3)
Requirement 4.1: Pgm builds set of localization hypothesis for current data
Validation 4.1: Test 4.1
Test 4.1: Run *generate_state_vector_pattern* process on program pre-state exported from corresponding Matlab process and compare post-state from Matlab output file to post-state of program.

Requirement 4.2: For any hypothesis in set from (4.1), pgm can compute support for that hypothesis
Validation 4.2: Test 4.2
Test 4.2: Run *compute_hypothesis_support* process on program pre-state exported from corresponding Matlab process and compare post-state from Matlab output file to post-state of program.

Requirement 4.3: Selected hypothesis support from set in (4.1) is used to update feature information
Validation 4.3: Test 4.3
Test 4.3: Run *ransac_hypothesis* process on program pre-state exported from corresponding Matlab process and compare post-state from Matlab output file to post-state of program.

Requirement 5: RANSAC hypothesis from (Req 4) is used to update feature mapping data
Requirement 5.1: RANSAC hypothesis from (Req 4) is used to update low-innovation inliers
Validation 5.1: Test 5.1 – dependent on Reqs (1-2)
Test 5.1: Run *ekf_update_li_inliers* process on program pre-state exported from corresponding Matlab process and compare post-state from Matlab output file to post-state of program.

Requirement 5.2: High-innovation inliers are identified from information retrieved from (5.1)
Validation 5.2: Test 5.2 – dependent on Reqs (1-2, 5.1)
Test 5.2: Run *rescue_hi_inliers* process on program pre-state exported from corresponding Matlab process and compare post-state from Matlab output file to post-state of program.

Requirement 5.3: Mapping information for high-innovation inlier features is updated
Validation 5.3: Test 5.3 – dependent on Reqs (1-2, 5.1, 5.2)
Test 5.3: Run *ekf_update_hi_inliers* process on program pre-state exported from corresponding Matlab process and compare post-state from Matlab output file to post-state of program.