Robot Pose Correction Using External Vision

1 Overview

This document describes the process for correcting robot positioning errors using external vision measurements. The system uses a Roboception camera (i.e., rc_visard or rc_viscore) to detect discrepancies between where the robot believes it is and where it actually is, then calculates a correction transformation that can be applied to future movements.

2 Process Description

2.1 Problem

- When the robot is commanded to move to a position ${}^{\text{cmd}}T_{\text{base}}$, it actually moves to a slightly different position ${}^{\text{actual}}T_{\text{base}}$ due to calibration errors, mechanical tolerances, and issues in the hand-eye-calibration.
- This discrepancy causes inaccuracies in precise operations like grasping or assembly.

2.2 Solution

1. **Estimate Robot Pose**: Using external vision (Roboception cameras and AprilTags), measure where the robot actually is (${}^{\text{actual}}T_{\text{base}}$) when it thinks it's at a commanded position (${}^{\text{cmd}}T_{\text{base}}$).
2. **Calculate Correction Transform**: Determine the homogeneous transformation matrix ${}^{\text{cmd}}T_{\text{actual}}$ that maps from desired target positions to the commands needed to reach those targets.
3. **Apply Correction**: Use this transformation matrix to adjust future movement commands to compensate for the robot's inherent errors.

2.3 Mathematical Representation

Let:

- ${}^{\text{cmd}}T_{\text{base}}$ = Position commanded to the robot (homogeneous transformation)
- ${}^{\text{actual}}T_{\text{base}}$ = Position where the robot actually ends up (measured by vision)
- ${}^{\text{cmd}}T_{\text{actual}}$ = The correction transformation matrix

During calibration, we measure both ${}^{\text{cmd}}T_{\text{base}}$ and ${}^{\text{actual}}T_{\text{base}}$, and determine ${}^{\text{cmd}}T_{\text{actual}}$ such that:

$$ {}^{\text{cmd}}T_{\text{base}} = {}^{\text{cmd}}T_{\text{actual}} \cdot {}^{\text{actual}}T_{\text{base}} \tag{1} $$

This is accomplished by multiplying both sides by the inverse of ${}^{\text{actual}}T_{\text{base}}$:

$$ {}^{\text{cmd}}T_{\text{actual}} = {}^{\text{cmd}}T_{\text{base}} \cdot ({}^{\text{actual}}T_{\text{base}})^{-1} \tag{2} $$

For future operations, to reach any target position ${}^{\text{target}}T_{\text{base}}$:

$$ {}^{\text{cmd}}T_{\text{base}} = {}^{\text{cmd}}T_{\text{actual}} \cdot {}^{\text{target}}T_{\text{base}} \tag{3} $$

2.4 Implementation Details

The calibration and correction process consists of these key steps:

2.4.1 Calibration Phase

- The robot moves to a known position (PrePointPose) where AprilTags are visible to the Roboception camera
- The system measures where the robot thinks it is (${}^{\mathrm{cmd}}T_{\mathrm{base}}$) using `P_cmd = $POS_ACT_MES : INV_POS($TOOL)`
- External vision determines where the robot actually is (${}^{\mathrm{actual}}T_{\mathrm{base}}$)
- The correction transformation is calculated using `T_correction = P_cmd :  INV_POS(P_actual)`
- This calculation gives us the transformation that converts from actual space to command space

2.4.2 Application Phase

- For any desired target position (${}^{\mathrm{target}}T_{\mathrm{base}}$), we calculate a corrected command position ${}^{\mathrm{corrected}}T_{\mathrm{base}}$
- This is done by applying `CorrectedPointPose = T_correction :  PointPose`
- When the robot moves to ${}^{\mathrm{corrected}}T_{\mathrm{base}}$, it will actually end up at the desired ${}^{\mathrm{target}}T_{\mathrm{base}}$
- The system effectively pre-compensates for the robot's systematic errors

2.4.3 Verification Phase

- The system demonstrates the correction by first moving to the uncorrected position (${}^{\mathrm{target}}T_{\mathrm{base}}$)
- Then it moves to the corrected position (${}^{\mathrm{corrected}}T_{\mathrm{base}}$)
- The user can visually confirm that the corrected position achieves better accuracy

In KRL robot programming language, the `:` operator represents frame multiplication, and the order matters. The correction works because we're using the same systematic error relationship that was measured during calibration to pre-compensate for future movements.

3 Usage

1. Run the `EstimateRobotPose` program
2. The robot will move to calibration positions
3. Vision system will measure actual positions
4. The correction transformation matrix is calculated
5. The robot will demonstrate both uncorrected and corrected movements
6. Apply this transformation matrix when planning robot movements for improved accuracy