Detailed Description of BEAST Protocol and Performance Indicators

1 Introduction

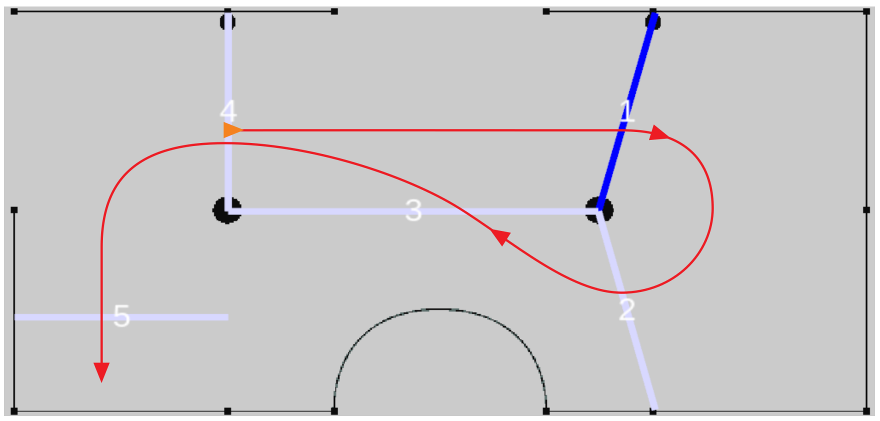

Figure 1: The layout of the arena. Orange: the start pose. Red: the path that the robot must follow with the trolley. Blue: the first checkpoint. Cyan: the remaining checkpoints.

In the following we refer to the trolley, but any consideration applies to the walker as well since the two are equivalent from the perspectives of the protocol and software.

The arena is composed by the walls, two big cylinders and two smaller cylinders. The cylinders are used to provide a visual marker to the robot and define the path that the robot must follow with the trolley. The path is defined by 5 checkpoints as shown in Fig. 1.

TODO: The pose is computed with the localization node (AMCL). The localization node provides an update of the pose when the trolley has moved 0.1m or rotated 0.25 radians (around 14 degrees) from the previous update.

2 Conditions

The benchmark has three condition variables: Disturbance type, Load and Start already gripping.

The *disturbance type* indicates the type of disturbance applied to the motion of the trolley using the actuated wheels (the front wheels for the trolley and the rear wheels for the walker). Possible values of *disturbance type* are *no force* and *sudden force*. When *no force* is selected, no braking force will be applied. When *sudden force* is selected, the actuated wheels will periodically apply a braking force on both wheels for a short duration of time (around 0.5 seconds). The braking force is applied every 15 radians of wheels rotation, which is equivalent to 1.23m when moving in a straight line. The wheels rotation is computed from the average amount of rotation from both wheels, meaning that the braking force is eventually applied no matter how the trolley is maneuvered (even if the cart is being rotated around one of the wheels).

The *load* indicates the amount of mass added inside the cart. Possible values are 0 and 9000 grams. We choose 9000g in order to use a pack of 6 1.5L water bottles, which are easy to get, store and handle.

When *start already gripping* is true, the robot starts the benchmark with its end effectors already positioned on the trolley's handle. When false, the robot must grip the handle by itself. We included this variable to allow executing a less complex variant of the benchmark that does not require the perception of the trolley and handle, end effector motion planning, etc.

3 Procedure

For the benchmark to start, the trolley must first be set in the start pose. Once this precondition is satisfied, the benchmark can start. In order to execute the benchmark, the robot must execute, in order, the following steps:

1. Grasp the trolley's handle. This is not required if the robot starts with its end effectors already positioned on the handle.
2. Navigate with the trolley through checkpoint 1.
3. Navigate with the trolley through checkpoint 2.
4. Navigate with the trolley through checkpoint 3.
5. Navigate with the trolley through checkpoint 4.
6. Navigate with the trolley through checkpoint 5.

The benchmark stops when manually interrupted by the referee via the GUI. Note that the pose of the trolley is considered as the center of the trolley's front wheels and not the pose of the robot operating the trolley, therefore we consider that the trolley has navigated through a checkpoint when the center of its actuated wheels crosses the checkpoint's line. We advice to move the trolley past the checkpoint line with some margin, because the localization may not always be accurate and if the trolley was stopped exactly on the checkpoint line, the event may not be detected. We suggest a margin of 0.3m past the checkpoint line.

4 Pre-Processed Data

4.1 Event

The event time series contains detectable events and their timestamps.

The possible events are:

- benchmark_start: identifies the start of benchmark execution. This event is always present. The timing of this event is affected by a delay due to the people operating the robot having to manually start the robot.
- handle_is_touched: identifies when the robot has gripped the handle. This is measured by observing the first time that the force applied to the handle exceeds a threshold.
- cart_starts_moving: identifies when the trolley has started moving from its initial pose. This is measured by observing when any of the two actuated wheels has rotated past a small threshold (0.1 radians).
- checkpoint_1..5.

The checkpoint events are computed by observing when the pose of the trolley crosses the checkpoint lines. The checkpoints can only be crossed once and in the correct order. If the robot moves the trolley through a checkpoint multiple times or in the wrong order, these occurrences are simply ignored. A checkpoint could be crossed in both directions, although this should not normally happen if the robot follows the path in Fig. 1. The checkpoints have been positioned in such a way that the shortest route from one checkpoint to the next results in the path.

4.2 Landmark Trajectory

This pre-processed data provides the position of the trolley as computed by the localization node (AMCL). The data contains two columns: time (ros time in seconds since epoch), cart_x [m], cart_y [m]. The coordinates of the position are in the frame of reference of the arena. The origin of this frame of reference corresponds to the starting pose of the trolley. The localization node provides an update of the pose when the trolley has moved 0.1m or rotated 0.25 radians (around 14 degrees) from the previous update.

4.3 Wrench

The data contains two columns: time (ros time in seconds since epoch), force_x $[N]$ The force applied on the handle is copied from the raw data from the ros topic /beast_cart/handle/force and converted to Newtons.

4.4 Distance

The data contains two columns: time (ros time in seconds since epoch), distance $[m]$ The distance is computed as the minimum range of each reading from the scan sensor. The scan sensor readings are filtered in order to ignore the ranges corresponding to the structure of the trolley and the legs of the robot. The filtered scans are published on the ros topic /beast_cart/scan_filtered.

5 Performance Indicators

5.1 Safety of Navigation

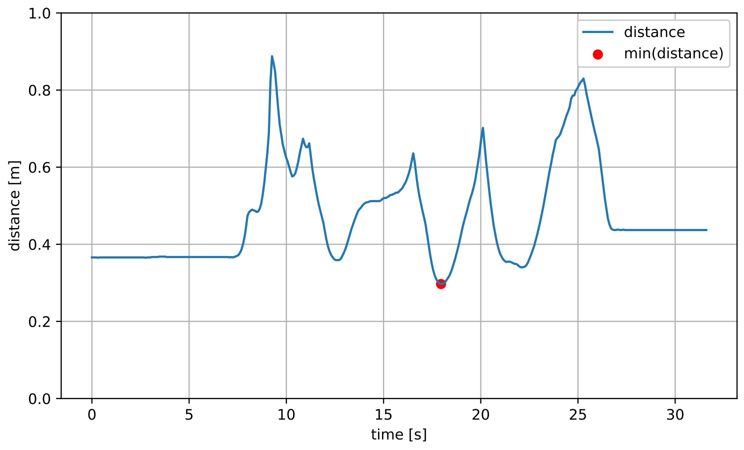

Figure 2: Plot of the distance of the sensor from the obstacles in the arena from the distance pre-processed data (blue) and its minimum value (red) used to compute the value of the PI. The data is collected in a test run in which a person operates the trolley.

This PI is a measurement of the safety of navigation of the robot based on the minimum distance from obstacles as measured by the scan sensor (using the distance pre-processed data). The value is computed as $min(x_i)$, where x_i are the values of distance in the distance pre-processed data. See Fig. 2. Note that this value corresponds to the distance of the closest obstacle to the lidar sensor, rather than the distance to the trolley.

5.2 Roughness of Actuation

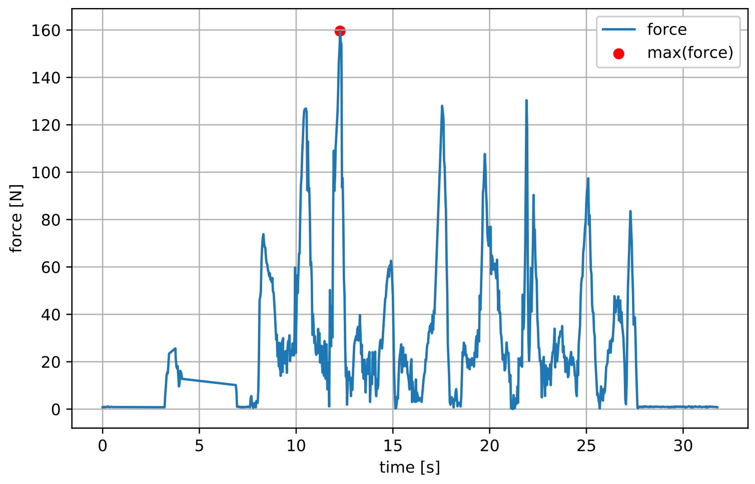

Figure 3: Plot of the force acting on the handle from the wrench pre-processed data (blue) and its maximum value (red) used to compute the value of the PI. The peaks correspond to the braking force applied by the actuated wheels. The data is collected in a test run in which a person operates the trolley.

TODO: update justification for beast.

This PI measures how rough (or undelicate) the robot is in operating the trolley based on the maximum force applied to the handle. High force applied to the handle is a symptom of bad handling of the trolley. The maximum force applied to the handle is affected by the braking force applied by the actuated wheels and by the load in the trolley. For this reason, the values of this PI can only be compared with runs that have the same disturbance type and load conditions. The value is computed as $max_i(|f_i|)$, where f_i are the values of force from the wrench pre-processed data.

5.3 Capability Level

Number of steps of the benchmark actually completed by the robot. Each step is considered completed only after all the steps preceding it have been completed as well. The result is an integer ranging from 0 to the number of steps composing the benchmark procedure. If the condition variable *starts already gripping* is false, then the number of steps is 6, otherwise the number of steps is 5 (since the step about grasping the handle is ignored).

5.4 Time to Grip Handle

This PI measures the amount of time between the start of the benchmark and when the robot touches the handle for the first time. The result is computed from the event pre-processed data using the events benchmark_start and handle_is_touched. If the condition variable *starts already gripping* is true, then the time is always 0 seconds. If the robot has not touched the handle the result of the PI is "TIMEOUT".

5.5 Time to Checkpoint n

These 5 PIs measures how long it took to reach each of the 5 checkpoints from the moment the trolley left the initial position. The result is computed from the event pre-processed data using the events cart_starts_moving and checkpoint_n, where n is the checkpoint number from 1 to 5. If the condition variable *starts already gripping* is false, meaning that the robot should grip the handle by itself, but the handle was not touched, then the result of the PI is "TIMEOUT" even if somehow the trolley crossed the checkpoint. This ensures the results of these PIs are consistent with the capability level PI. If the trolley has not crossed the checkpoint n, the result of the PI is "TIMEOUT".