Collaborative Robotics Modeling

FM

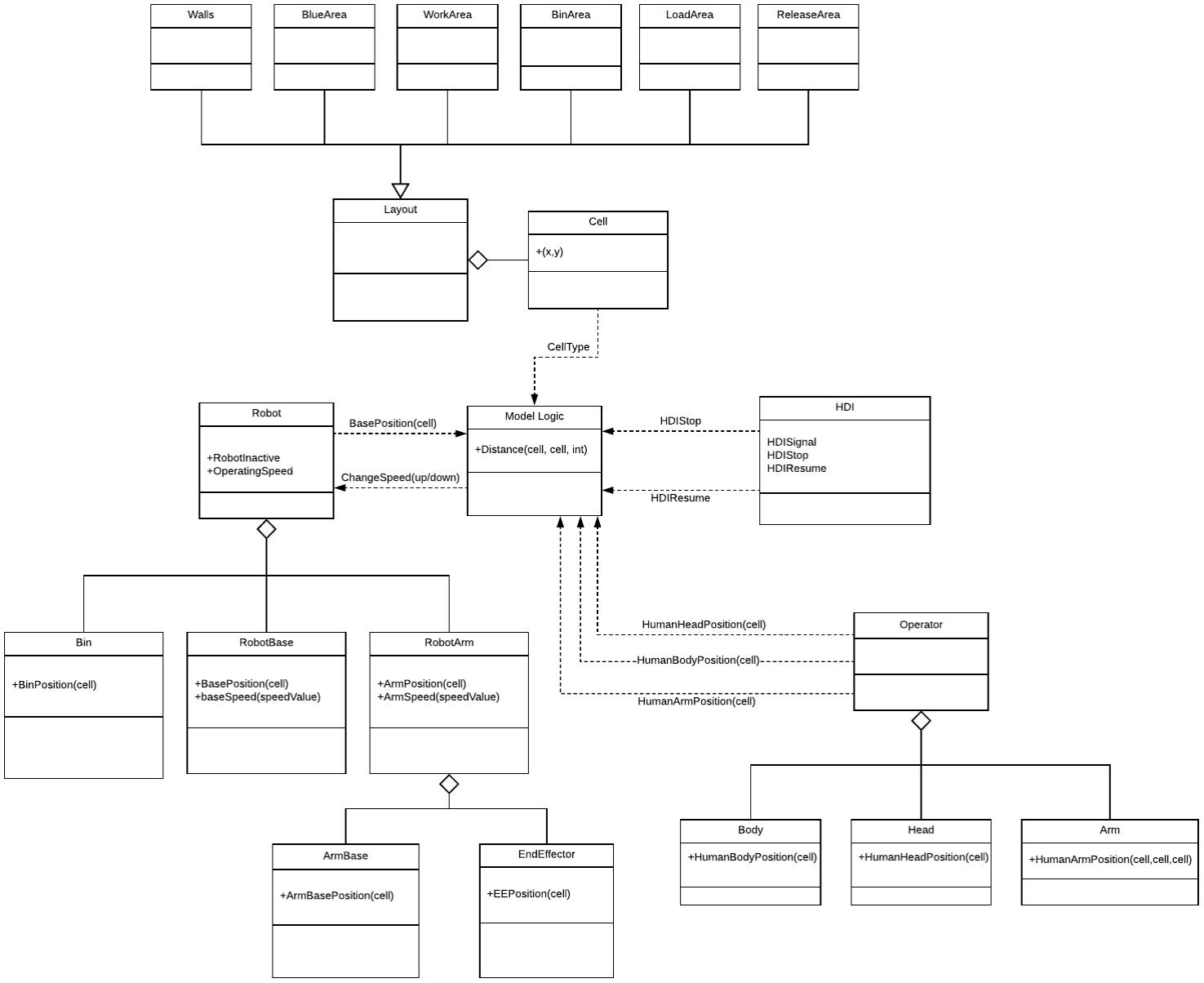
CS 2018

Davide Santambrogio
Federico Sandrelli
Andrea Tricarico

Sommario

UML

Here is the general structure of our system: the main classes and how they interact.



Axioms

Cell

NOTE ON CELLS
This model is based on “cells”. Cells identify a 20cm x 20cm area and they represent the granularity of our sensor. The layout is divided in cells and the position of every object that enters the scene is defined by the cells it occupies. Movements are therefore discrete.

Throughout this project, cells will be identified in two ways:

- With its coordinates on the 2D plane (for when we need the coordinates): (x, y)
- With a variable: $cell_i,\ cell_j,\ etc...$

So, for example:

- $Alw(BasePosition(cell_i) \Rightarrow BlueArea(cell_i))$
- $Alw(BasePosition((x, y)) \Rightarrow BlueArea((x, y)))$

Are two equivalent axioms. They are saying that if the position of the base is detected on $cell_i$ $((x, y))$ (generic cell), then $cell_i$ ((x, y)) must be one of the cells in the blue area.

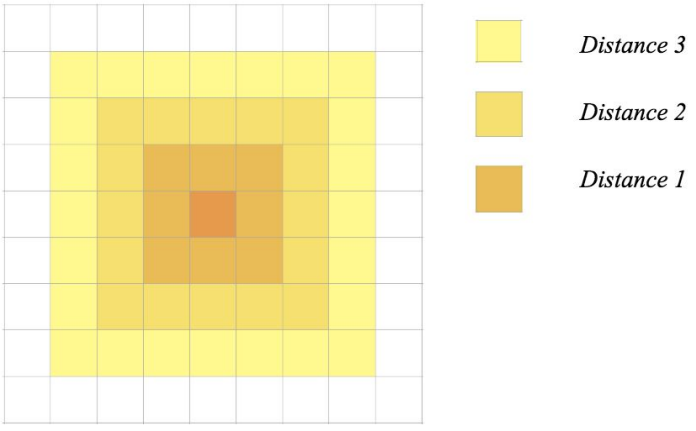
Distance((x,y),(w,z), d)

This predicate checks whether the two cells, of which the coordinates are specified, are [d] cells apart from each other (i.e. there are [d-1] cells between them). This distance measure is an approximation that measures distance with a square perimeter. We have chosen to calculate distance this way because it fits well with the how movements have been modelled.

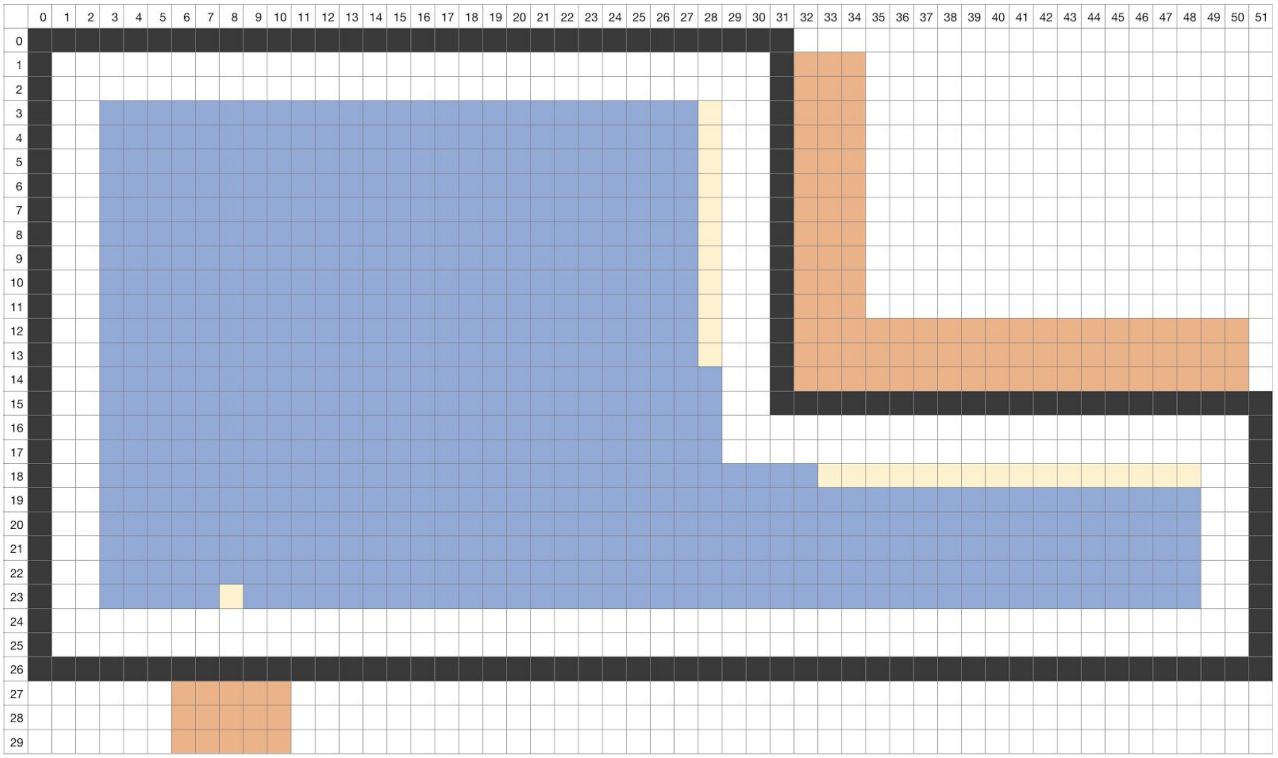
k1.

$$Alw(\ \forall x,y,w,z,d\ (Distance((x,y),(w,z),d)\ \Leftrightarrow d\ =\ max(abs(x-w),\ abs(y-z)))$$

Note that “Distance(cell1, cell2, 0)” means that cell1 and cell2 are the same cell.



Layout



Walls((x,y))

This predicates delimits the area of the walls, that delimit the area in which the robot can operate.

k2.

$$Alw(\ \forall x,y\ (Walls(x,y)\ \Leftrightarrow\ (\ (x=0\ \wedge\ 0\le y\le 26)\ \vee\ (1\le x\le 31\ \wedge\ 0\le y\le 15)\ \vee\ (\ 32\le x\le 51\ \wedge\ y=15)\ \vee\ (x=51\ \wedge\ 16\le y\le 26)\ \vee\ (\ 1\le x\le 50\ \wedge\ y=26)\ ))\ )$$

BlueArea((x,y))

This predicates delimits the area of the blue cells. These are the allowed positions for the center of the robot's base. It will be more clear after reading the "Robot" chapter.

k3.

$$Alw(\ \forall x,y\ (BlueArea(x,y)\ \Leftrightarrow\ (\ (2\le x\le 28\ \wedge\ 3\le y\le 23)\ \vee\ (29\le x\le 48\ \wedge\ 18\le y\le 23)))\ )$$

WorkArea((x,y))

k4.

$$Alw(\ \forall x,y\ (WorkArea(x,y)\ \Leftrightarrow\ (6\le x\le 10\ \wedge\ 27\le y\le 29)\ )\ )$$

BinArea((x,y))

k5.
$Alw(\ \forall x,y\ (BinArea(x,y)\ \Leftrightarrow\ (32 \le x \le 34\ \wedge\ 1 \le y \le 14)\ \vee\ (35 \le x \le 50\ \wedge\ 12 \le y \le 14)\ )$

LoadArea(x,y)

k6.
$Alw(\ \forall x,y\ (LoadArea(x,y)\ \Leftrightarrow\ (x = 28\ \wedge\ 3 \le y \le 13) \vee (33 \le x \le 48\ \wedge\ y = 18)\ )$

ReleaseArea(x,y)

k7.
$Alw\ (\forall x,y\ (ReleaseArea(x,y)\ \Leftrightarrow\ (x = 8\ \wedge\ \ y = 23)\ )$

HDI

The Human Device Interface is a handheld device that allows the operator to pause the robot's workflow ad resume it at any moment.

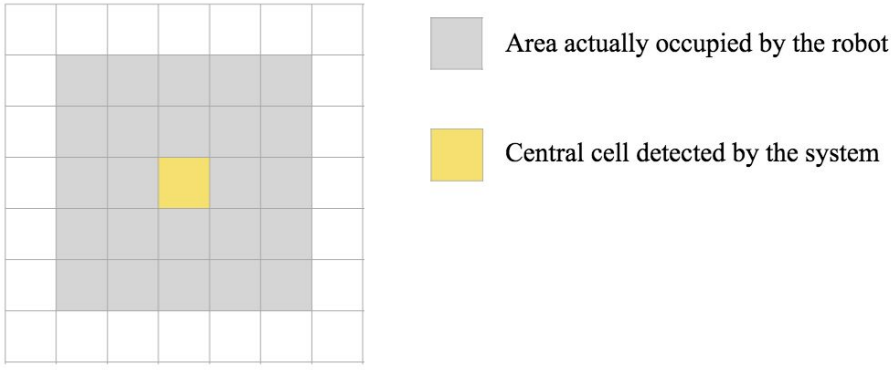
k8.
$HDISignal \Leftrightarrow HDIStop \vee HDIResume$

k9.
$HDIResume \Rightarrow SomeP(HDIStop)$

k10.
$HDISignal \Rightarrow UpToNow(\neg HDISignal) \wedge NowOn(\neg HDISignal)$

"HDISignal", "HDIStop", "HDIResume" are signals. They will be further explained in the 'Signals' chapter.

Robot



BasePosition

This predicate assures that the robot always has a position and this is identified by a unique cell (it represents the cell occupied by the center of the robot's base). In the scenario of this project we only have one robot.

k11.
$Alw(\exists !cell_i(BasePosition(cell_i)))$

Operating Speed

OperatingSpeed is a global variable used by ChangeSpeed, this variable represents the current speed of the robot intended as the current speed of the Base, in the case in which the Robot is in mobile mode, or as the current speed of the arm, in the case in which the Robot is in manipulator mode.

k12.
$(operatingSpeed = high) \Leftrightarrow$
$(Mode(mobile) \wedge BaseSpeed(high)) \vee (Mode(manipulator) \wedge ArmSpeed(high))$

k13.
$(operatingSpeed = medium) \Leftrightarrow$
$(Mode(mobile) \wedge BaseSpeed(medium)) \vee (Mode(manipulator) \wedge ArmSpeed(medium))$

k14.
$(operatingSpeed = low) \Leftrightarrow$
$(Mode(mobile) \wedge BaseSpeed(low)) \vee (Mode(manipulator) \wedge ArmSpeed(low))$

k15.
$(operatingSpeed = still) \Leftrightarrow$
$(Mode(mobile) \wedge BaseSpeed(still)) \vee (Mode(manipulator) \wedge ArmSpeed(still))$

ChangeSpeed(up/down) Event

ChangeSpeed is a signal modelled as an event, that lets the robot know it should change its operating speed. (NOTE: ChangeSpeed is an event, and therefore, taken a certain pair of values for operstingSpeed and accelerationDirection variables, it is instantaneous)

k16.

$$\forall operatingSpeed, accelerationDirection(ChangeSpeed(operatingSpeed, accelerationDirection) \Rightarrow UpToNow\ (\neg ChangeSpeed(operatingSpeed, accelerationDirection)) \wedge NowOn(\neg ChangeSpeed(operatingSpeed, accelerationDirection)))$$

Indeed, considering for instance operatingSpeed = high and accelerationDirection = down, ChangeSpeed(high,down) cannot be true for two consecutive instantans. However, it is possible to have ChangeSpeed(high,down) true at a certain time t and ChangeSpeed(medium,down) true at t+1.

Base Speed Change

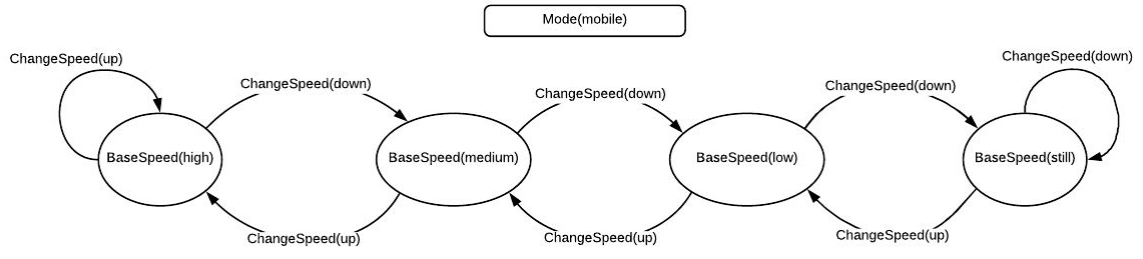
These axioms model the change of speed in the robot's movements. When in Mobile mode, only the robot's base will move, for this reason the ChangeSpeed event will affect only the BaseSpeed.

(NOTE: taken α as the Reaction Time parameter, the Robot will effectively change its speed α time-instants after the ChangeSpeed event arises).

In order to achieve correctly the Safety Property (as we will discuss later), the acceleration can be effective only if there are no CloseTo or SlowDown signals. Otherwise, the Robot will keep the same operating Speed for at least α instants.

k17.

$$\forall operatingSpeed,\ \exists d(Mode(mobile) \wedge ChangeSpeed(operatingSpeed, up) \wedge (Futr(ChangeSpeed(operatingSpeed, down), d) \vee Futr(CloseTo), d) \wedge d < \alpha \Rightarrow Lasts_{ii}(BaseSpeed(operatingSpeed, \alpha)))$$



These are all the formalized speed changes of the BaseSpeed:

k18.

$$\neg RobotInactive \wedge Mode(mobile) \wedge ChangeSpeed(high, up) \wedge$$
$$Lasts_{ie}(\neg ChangeSpeed(high, down) \wedge \neg CloseTo, \alpha) \Rightarrow$$
$$Lasts_{ie}(BaseSpeed(high, \alpha) \wedge Futr(BaseSpeed(high), \alpha)$$

k19.

$$\neg RobotInactive \wedge Mode(mobile) \wedge ChangeSpeed(medium, up) \wedge$$
$$Lasts_{ie}(\neg ChangeSpeed(medium, down), \alpha) \Rightarrow$$
$$Lasts_{ie}(BaseSpeed(medium, \alpha) \wedge Futr(BaseSpeed(high), \alpha)$$

k20.

$$\neg RobotInactive \wedge Mode(mobile) \wedge ChangeSpeed(low, up) \wedge$$
$$Lasts_{ie}(\neg ChangeSpeed(low, down), \alpha) \Rightarrow$$
$$Lasts_{ie}(BaseSpeed(low, \alpha) \wedge Futr(BaseSpeed(medium), \alpha)$$

k21.

$$\neg RobotInactive \wedge Mode(mobile) \wedge ChangeSpeed(still, up) \wedge$$
$$Lasts_{ie}(\neg ChangeSpeed(still, down), \alpha) \Rightarrow$$
$$Lasts_{ie}(BaseSpeed(still, \alpha) \wedge Futr(BaseSpeed(low), \alpha)$$

k22.

$$Mode(mobile) \wedge ChangeSpeed(high, down) \Rightarrow$$
$$Lasts_{ie}(BaseSpeed(high, \alpha) \wedge Futr(BaseSpeed(medium), \alpha)$$

k23.

$$Mode(mobile) \wedge ChangeSpeed(medium, down) \Rightarrow$$
$$Lasts_{ie}(BaseSpeed(medium, \alpha) \wedge Futr(BaseSpeed(low, \alpha)$$

k24.

$$Mode(mobile) \wedge ChangeSpeed(low, down) \Rightarrow$$
$$Lasts_{ie}(BaseSpeed(low, \alpha) \wedge Futr(BaseSpeed(still), \alpha)$$

k25.

$$Mode(mobile) \wedge ChangeSpeed(still, down) \Rightarrow$$
$$Lasts_{ie}(BaseSpeed(still, \alpha) \wedge Futr(BaseSpeed(still), \alpha)$$

Operating Mode - Mode(mobile)

This predicate tells us that, when located in the blue area outside of the LoadArea or the ReleaseArea, the robot should be operating in "mobile" stateType. This means that only the robot's body is functioning (to move around), while the arm is in the rest position to improve safeness of movements.

k26.

$$Alw((BasePosition(cell_i) \wedge \neg(LoadArea(cell_i) \vee ReleaseArea(cell_i))) \Rightarrow Mode(mobile))$$

The moment in which the Robot switch its mode to mobile is when it is in the LoadArea with BinFull or in the ReleaseArea with BinEmpty.
NOTE: BinEmpty and BinFull are two states

k27.

$$UpToNow(Mode(manipulator) \wedge ArmInRestPosition \wedge BasePosition(cell_i) \wedge ((BinFull \wedge LoadArea(cell_i)) \vee (BinEmpty \wedge ReleaseArea(cell_i))) \vee \Leftrightarrow Becomes(Mode(mobile))$$

There are only two mode for the Robot and one excludes the other

k28.

$$Alw(Mode(mobile) \Leftrightarrow \neg Mode(manipulator))$$

k29.

$$Alw(Mode(mobile) \Rightarrow ArmInRestPosition)$$

Operating Mode - Mode(manipulator)

At any instant the robot can be operating in only one of the following two states: Mobile, Manipulator. When it is in the manipulator mode, the robot's base should be still and the arm should be operating on a task (picking up or putting down the WPs).

k30.

$$Alw(Mode(manipulator) \Rightarrow BaseSpeed(still))$$

The moment in which the Robot switch its mode to manipulator is when it is in the LoadArea with BinEmpty or in the ReleaseArea with BinFull.

k31.

$$UpToNow(Mode(mobile) \wedge ArmInRestPosition \wedge BasePosition(cell_i) \wedge ((BinEmpty \wedge LoadArea(cell_i)) \vee (BinFull \wedge ReleaseArea(cell_i))) \Leftrightarrow Becames(Mode(manipulator))$$

RobotInactive

The robot becomes “inactive” when it receives the “HDIStop” signal. When the robot enters this state it slows down (base and arm) until it is completely still and remains still until it receives the “HDIResume” signal. Once it exits the Inactive state, it will resume its activities autonomously.

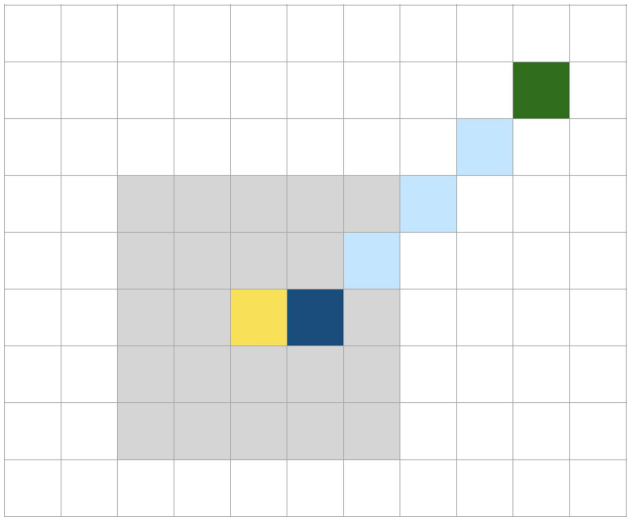
k32.
$HDIStop \Rightarrow RobotInactive$

k33.
$RobotInactive \Leftrightarrow Since(\neg HDIResume, HDIStop)$

Arm

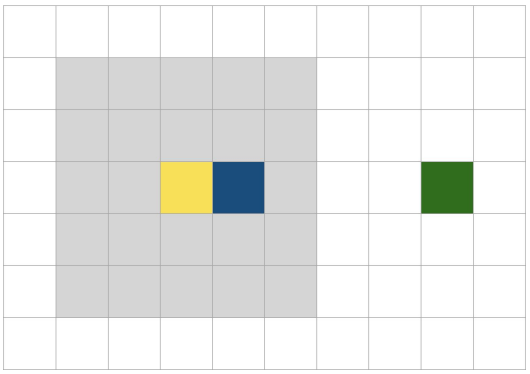
The arm of the robot is modelled by the ArmBase and the EndEffector. These two components are on the opposite extremities of the arm, therefore it is sufficient to consider only these two elements, knowing that the cells connecting them (in a straight row) are also occupied by the arm.
This simplification will allow us to simplify our axioms while still keeping all the information required to ensure safety.
The fully extended arm can occupy a straight row of 5 cells (total of 1 meter).

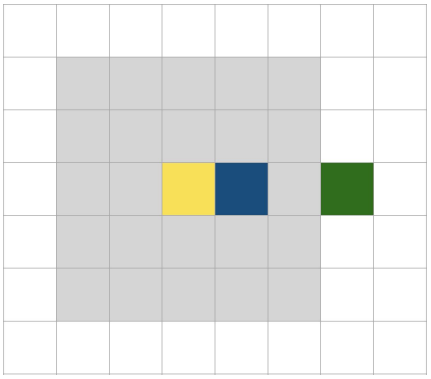
APPROXIMATION NOTE:
The robot arm is modeled as a row of five cells that can also be diagonally aligned. We realize that the diagonal length is longer than the vertical or horizontal length, but we have decided to accept this approximation of the model.



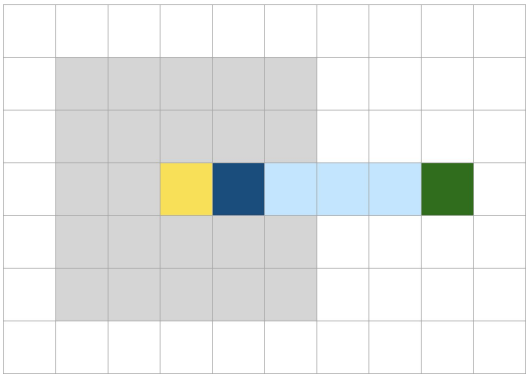
The following two images show how the bending of the arm results in a contraction of the area occupied by it. Only detecting the End Effector is very effective in modelling this phenomenon in a simple but precise way.

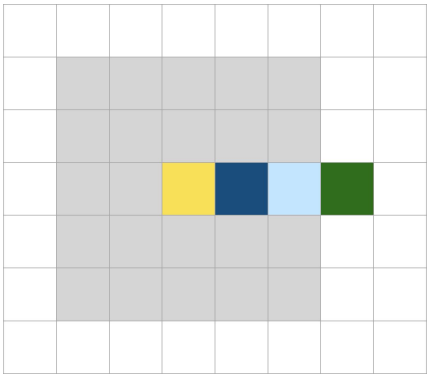
What the system sees:

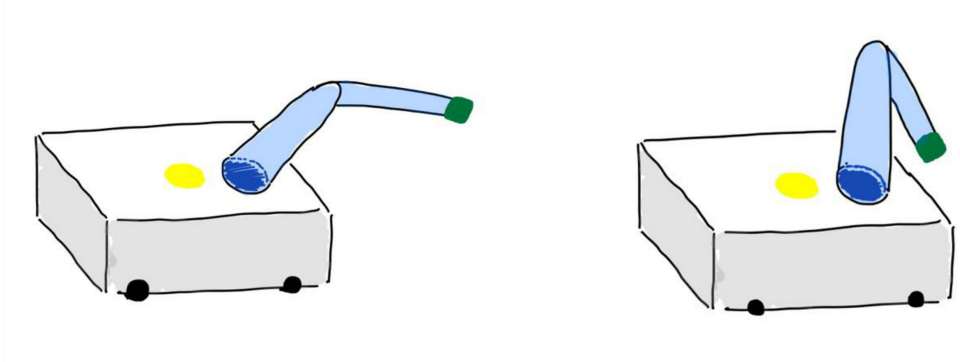




What we know is happening:







ArmPosition($cell_i$, $cell_j$)

The two cells of the ArmPosition predicate represent, respectively, the positions of the ArmBase and of the EndEffector.

k34.

$$Alw(\forall cell_i, cell_j(ArmPosition(cell_i, cell_j) \Leftrightarrow ArmBasePosition(cell_i) \wedge EEPosition(cell_j)))$$

There is Only One Arm

In the scenario of this project we only have one robot, therefore only one robot arm as well.

k35.

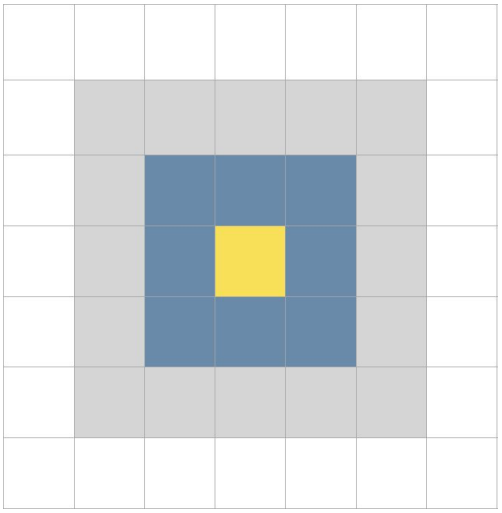
$$Alw(\exists !cell_i, cell_j(ArmPosition(cell_i, cell_j)))$$

ArmBase Movement Area

The ArmBase represents the point of attachment between robot and arm. To model the rotation of the robot, the arm base can change position around the robot's BasePosition.

k36.

$$Alw(\forall cell_i, cell_j, cell_k(ArmPosition(cell_i, cell_j) \wedge BasePosition(cell_k) (ArmBasePosition(cell_i) \wedge Distance(cell_i, cell_k, 1))))$$



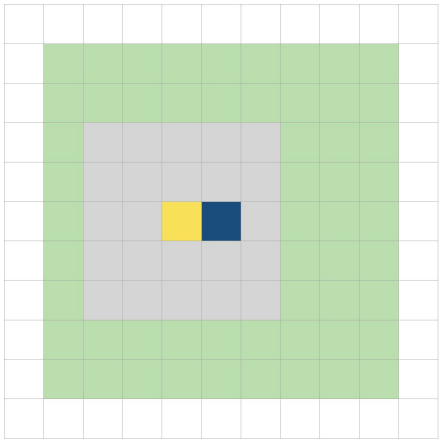
This is the area in which the ArmBase can move (blue), given the position of the robot's BasePosition (yellow).
Note that as the arm base moves, the reach of the EndEffector changes.

EndEffector Movement Area

Here we describe the movement limitations that the EE must comply to when the robot is operating in "manipulator" mode.
Since the Arm Base is not going to move in manipulator mode, we just need to limit the relative distance of the End Effector with the respect to the base (not over 5 cells).

k37.

$$Alw(\forall cell_i,\ cell_j,\ v($$
$$ArmPosition(cell_i,\ cell_j)\ \Rightarrow$$
$$\exists v((v > 0\ \wedge v < 5) \wedge Distance(cell_i,\ cell_j,\ v))))$$



This is the full area (light green) in which the End Effector can move (the Robot's Base is included), given the position of the ArmBase (dark blue).

ArmSpeed - Mode(Manipulator)

Here we define the speed of the arm when the robot is operating in "Manipulator mode". Given the nature of the arm as seen from the model (straight long object, rotating with center of rotation on the ArmBasePosition) we know that the EE will be the fastest component during arm movement. Therefore we will only consider the EE for speed limitation.

k38.

$$Alw(Mode(Manipulator) \Rightarrow \forall cell_i,\ cell_j\ ($$
$$(EEPosition(cell_i)) \wedge ArmSpeed(low) \Rightarrow$$
$$Futr(EEPosition(cell_j), 1) \wedge (Distance(cell_i,\ cell_j,\ 1)))$$

k39.

$$Alw(Mode(Manipulator) \Rightarrow \forall cell_i,\ cell_j\ ($$
$$(EEPosition(cell_i)) \wedge ArmSpeed(medium) \Rightarrow$$
$$Futr(EEPosition(cell_j), 1) \wedge (Distance(cell_i,\ cell_j,\ 2)))$$

k40.

$$Alw(Mode(Manipulator) \Rightarrow \forall cell_i,\ cell_j\ ($$
$$(EEPosition(cell_i)) \wedge ArmSpeed(high) \Rightarrow$$
$$Futr(EEPosition(cell_j), 1) \wedge (Distance(cell_i,\ cell_j,\ 3)))$$

NOTE:
You can see that "ArmSpeed(still)" is missing; this is because here we are describing the speed of the arm only for when the robot is operating in Manipulator mode. "ArmSpeed(still)" will be specified separately since it applies to both manipulator and mobile mode.

ArmSpeed(still)

This predicate makes sure that, when the arm speed is "still", the robot's arm is not moving with respect to the robot's body.

k41.

$$Alw(\forall x, y, v, w,\ a,\ b,\ c,\ d($$
$$BasePosition(x, y) \wedge Futr(BasePosition(x + v,\ y + w), 1) \wedge ArmSpeed(still) \wedge$$
$$(ArmPosition((a_1, b_1), (c_1, d_1)))$$
$$\wedge Futr(ArmPosition((a_2, b_2), (c_2, d_2)),\ 1) \Rightarrow$$
$$(a_2 = a_1 + v) \wedge (c_2 = c_1 + v) \wedge$$
$$(b_2 = b_1 + w) \wedge (d_2 = d_1 + w))$$
$$\vee$$
$$RobotRotation$$

What this is saying:
The first part is saying that, at every time instant, all cells of the arm have moved exactly as the robot's base has moved. The last clause is to take into account the case in which the robot is rotating.

Arm in Rest Position

At any instant the robot can be operating in only one of the following two states:

- Mobile
- Manipulator.

When it is moving around (i.e. it is operating in the Mobile state), the arm should not protrude from the robot's body and it should not be moving. We can model this by simply assuring that the EE is within the robot's base area.

k42.

$$Alw($$
$$ArmInRestPosition \Leftrightarrow (ArmSpeed(still) \wedge \neg \exists cell_i,\ cell_j,\ cell_k($$
$$ArmPosition(cell_i,\ cell_j) \wedge BasePosition(cell_k) \wedge$$
$$(\exists v(v > 2 \wedge Distance(cell_k,\ cell_j,\ v)))$$

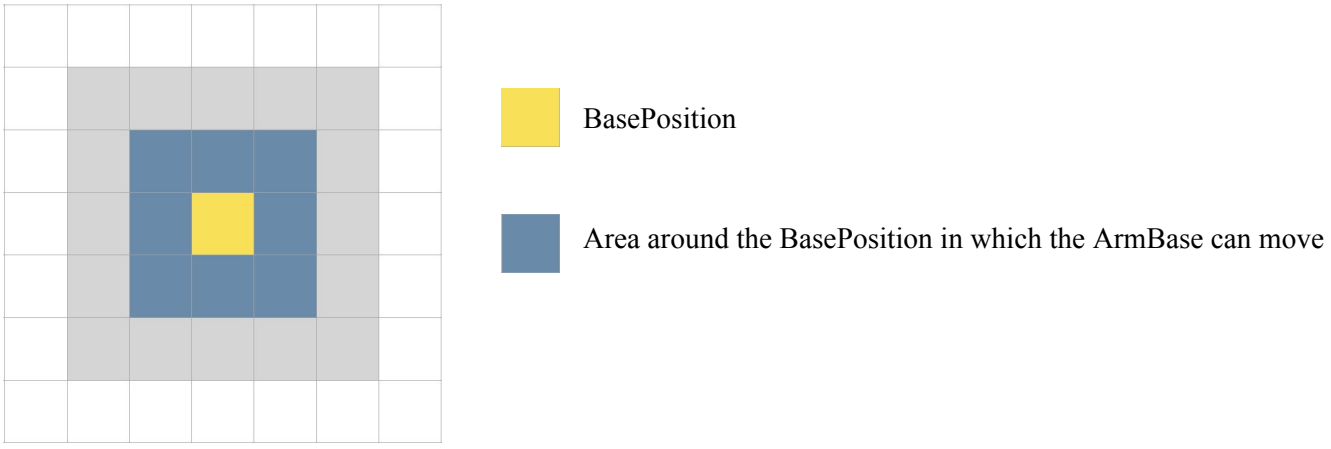
What this is saying:
None of the cells occupied by the arm are at a distance greater than 2 from the robot central cell.

RobotRotation

This predicate models the rotation of the robot. This is done by detecting the movement of the ArmBasePosition around the robot's central base cell.

k43.

$$Alw(Mode(mobile) \wedge RobotRotation \Rightarrow$$
$$BaseSpeed(still) \wedge$$
$$\exists cell_i,\ cell_j(Dist(cell_i, cell_j, 1) \wedge ArmBasePosition(cell_i) \wedge Futr(ArmBasePosition(cell_j,\ 1)))$$



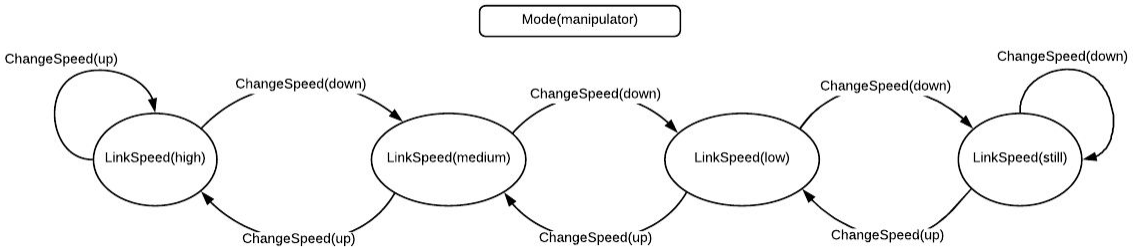
What this is saying:
When rotating, the center of the base remains still since it is the center of rotation while the rest of the body rotates around it. We are not detecting the rest of the robot's body but only the arm. Therefore, as the robot rotates, the arm will move with it and in particular, the arm's base.

The “ArmBasePosition” axiom ensures that the new position of the arm base is still next to the robot’s base.

APPROXIMATION NOTE:
We are saying nothing on the behaviour of the EE in this situation. The EE will still be limited to stay within the bounds of the robot because it is in Mobile mode so we think this approximation will not add any particular risks.

Arm Speed Change

These axioms model the change of speed in the arm’s movements. They are equal to the Base Speed Change’s axioms, with the only difference that the mode of the Robot must be manipulator.



k44.
$\forall operatingSpeed,\ \exists d(Mode(mobile) \wedge ChangeSpeed(operatingSpeed, up) \wedge$
$(Futr(ChangeSpeed(operatingSpeed, down), d) \vee Futr(CloseTo), d) \wedge d < \alpha \Rightarrow$
$Lasts_{ii}(BaseSpeed(operatingSpeed, \alpha)))$

k45.
$\neg RobotInactive \wedge Mode(manipulator) \wedge ChangeSpeed(high, up) \wedge$
$Lasts_{ie}(\neg ChangeSpeed(high, down), \alpha) \Rightarrow$
$Lasts_{ie}(ArmSpeed(high, \alpha)) \wedge Futr(ArmSpeed(high), \alpha)$

k46.
$\neg RobotInactive \wedge Mode(manipulator) \wedge ChangeSpeed(medium, up) \wedge$
$Lasts_{ie}(\neg ChangeSpeed(medium, down), \alpha) \Rightarrow$
$Lasts_{ie}(ArmSpeed(medium, \alpha)) \wedge Futr(ArmSpeed(high), \alpha)$

k47.
$\neg RobotInactive \wedge Mode(manipulator) \wedge ChangeSpeed(low, up) \wedge$
$Lasts_{ie}(\neg ChangeSpeed(low, down), \alpha) \Rightarrow$
$Lasts_{ie}(ArmSpeed(low, \alpha)) \wedge Futr(ArmSpeed(medium), \alpha)$

k48.
$\neg RobotInactive \wedge Mode(manipulator) \wedge ChangeSpeed(still, up) \wedge$
$Lasts_{ie}(\neg ChangeSpeed(still, down), \alpha) \Rightarrow$

$$Lasts_{ie}(ArmSpeed(still, \alpha)) \wedge Futr(ArmSpeed(low), \alpha)$$

k49.
$$Mode(manipulator) \wedge ChangeSpeed(high, down) \Rightarrow$$
$$Lasts_{ie}(ArmSpeed(high, \alpha)) \wedge Futr(ArmSpeed(medium), \alpha)$$

k50.
$$Mode(manipulator) \wedge ChangeSpeed(medium, down) \Rightarrow$$
$$Lasts_{ie}(ArmSpeed(medium, \alpha)) \wedge Futr(ArmSpeed(low), \alpha)$$

k51.
$$Mode(manipulator) \wedge ChangeSpeed(low, down) \Rightarrow$$
$$Lasts_{ie}(ArmSpeed(low, \alpha)) \wedge Futr(ArmSpeed(still), \alpha)$$

k52.
$$Mode(manipulator) \wedge ChangeSpeed(still, down) \Rightarrow$$
$$Lasts_{ie}(ArmSpeed(still, \alpha)) \wedge Futr(ArmSpeed(still), \alpha)$$

Bin

BinPosition

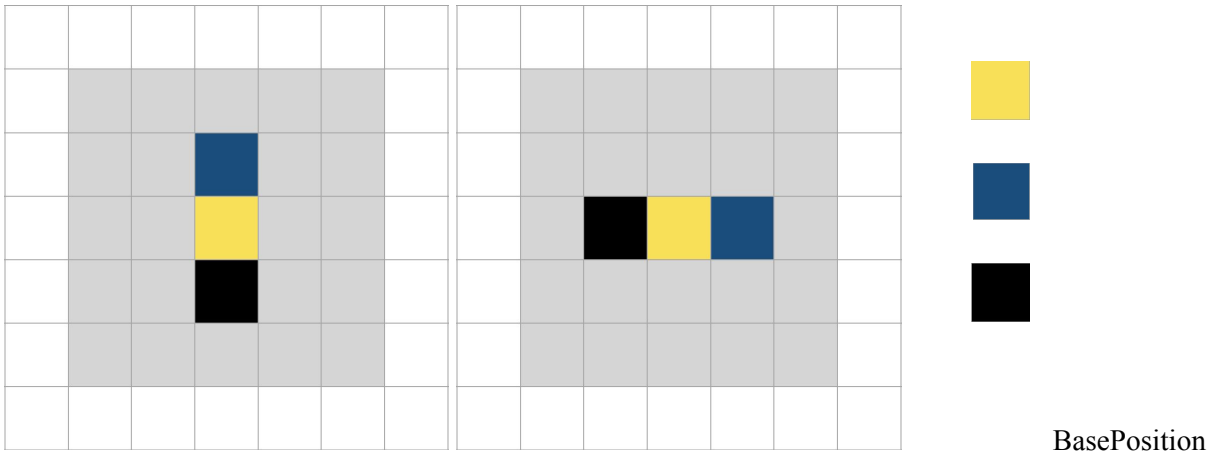
We have decided to model the bin to have the dimensions of a single cell on the robot's body. It is located on the side opposite to the arm base (the arm base is the first cell of link1). The motion of the arm base simulates rotation

of the robot, so the bin will obviously rotate with it, remaining on the opposite side of the arm base as it moves around the robot base's center.

k53.

$$Alw($$
$$(BinPosition((x,y)) \wedge BasePosition((w,z))) \wedge ArmBasePosition((s,t)) \Rightarrow$$
$$(\exists a,b(a,b \in \{-1,0,1\} \wedge$$
$$(x = w - a) \wedge (y = z - b) \wedge (s = w + a) \wedge (y = z + b)))$$

In the following images we can see an example of how the bin follows the arm base symmetrically.



BasePosition

ArmBasePosition

BinPosition

There is Only one Bin

k54.

$$Alw(\exists !cell_i(BinPosition(cell_i)))$$

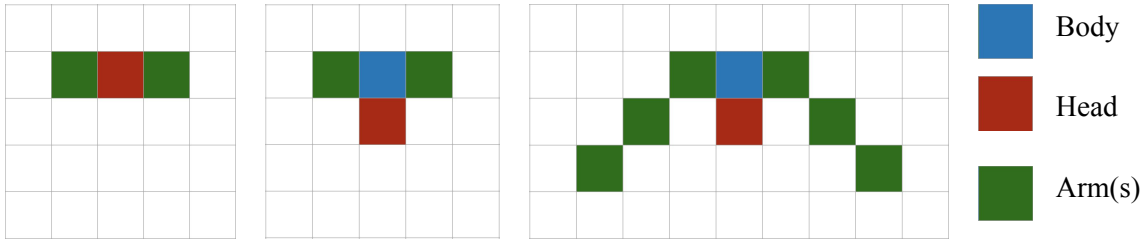
Operator

The operator is composed of a body (1 cell), a head (1 cell) and two arms (can occupy 1 to 3 connected cells).
The head can be directly over the body or next to it.
The arms are attached to the body and therefore the first cell they occupy is always next to the body.

The three images represent (from left to right), respectively, the operator (looking toward south):

- Standing straight with arms along waist
- Leaning forward
- Leaning forward with arms fully extended forward

(many more positions are possible, these are just a few relevant examples)



NOTE: We are not modelling any constraint on the number of operators or on their behaviour, to be as resistant as possible to unpredictable human behaviours.

Operator Position

This predicate describes the structure of an operator.

k55.

$$Alw($$
$$\exists cell_i(HumanBodyPosition(cell_i)) \Rightarrow$$
$$\exists cell_j, cell_k, cell_l, cell_m, cell_n, cell_o, cell_p(HumanHeadPosition(cell_j) \wedge$$
$$HumanArmPosition(cell_k,\ cell_l,\ cell_m) \wedge HumanArmPosition(cell_n,\ cell_o,\ cell_p) \wedge$$
$$(cell_i = cell_j \vee Distance(cell_i,\ cell_j,\ 1)) \wedge$$
$$(Distance(cell_i,\ cell_k,\ 1) \wedge Distance(cell_i,\ cell_n,\ 1))$$

Cells Occupied by a HumanArm are Connected

Here we ensure that an operator's arm occupies at most 3 connected cells.

k56.

$$Alw(\forall cell_i,\ cell_j,\ cell_k($$
$$HumanArmPosition(cell_i,\ cell_j,\ cell_k) \Rightarrow \exists v((v \in \{0,\ 1\}) \wedge$$
$$Distance(cell_i,\ cell_j,\ v) \wedge Distance(cell_j,\ cell_k,\ v))))$$

Robot-Layout Interaction

Robot Body Always in Blue Area

k57.

$$Alw(\forall cell_i(\ BasePosition(cell_i)\ \Rightarrow\ BlueArea(cell_i)))$$

Base Speed

These three axioms model the movement of the robot. They differentiate between three different speeds checking that the robots movement are coherent with the speed it should be moving at. Indeed, the Robot will cover from 0 to 3 cells in 1 instant depending on its Speed (from still to high).

k58.
$$BasePosition(cell_i) \wedge BaseSpeed(still)) \Rightarrow$$
$$Futr(BasePosition(cell_i), 1)$$

k59.
$$BasePosition(cell_i) \wedge BaseSpeed(low)) \Rightarrow$$
$$Futr((\ BasePosition(cell_j) \wedge Distance(cell_i, cell_j,\ 1)\ ), 1)$$

k60.
$$BasePosition(cell_i) \wedge BaseSpeed(medium)) \Rightarrow$$
$$Futr((\ BasePosition(cell_j) \wedge Distance(cell_i, cell_j,\ 2)\ ), 1)$$

k61.
$$BasePosition(cell_i) \wedge BaseSpeed(high)) \Rightarrow$$
$$Futr((\ BasePosition(cell_j)\ \wedge Distance(cell_i, cell_j,\ 3)\ ), 1)$$

APPROXIMATION NOTE
We are assuming that if the system detects the robots base in a cell and in the following instant it detects it occupying another cell, then in the time in between the two instants, the robot has moved along the shortest path connecting the two cells.

Wall's Constraint

The position of the robot's base has to fulfill the physical constraints given by the walls.

k62.
$$Alw(\forall cell_i,\ cell_j((BasePosition(cell_i)\ \wedge Wall(cell_j))\ \Rightarrow$$
$$\neg \exists d((d < 4) \wedge (Distance(cell_i, cell_j, d))))$$

Robot-Operator Interaction

Signal

This is the generic signal. It allows the robot to realise there is an exception to be handled and interrupt its routine in order to manage the issue recognized by the signal.

k63.

$$Signal \Leftrightarrow HDISignal \vee CloseTo$$

HDI Signals

These signals have the highest priority. They allow the operator to completely stop the robot in any situation and to reactivate it later on.

k64.

$$HDISignal \Leftrightarrow HDIStop \vee HDIResume$$

k65.

$$\neg RobotInactive \wedge HDIStop \Rightarrow Becomes(RobotInactive)$$

k66.

$$RobotInactive \wedge HDIResume \Rightarrow Becomes(\neg RobotInactive)$$

k67.

$$RobotInactive \wedge$$
$$\forall accelerationDirection(Lasted_{ee}(\neg ChangeSpeed(operatingSpeed, accelerationDirection), \alpha) \wedge$$
$$(\neg BaseSpeed(still) \vee \neg ArmSpeed(still)) \Rightarrow$$
$$ChangeSpeed(down)$$

CloseTo Signals

For the Proximity Signals we use the signals state “CloseTo” which becomes true when the Operator is within a certain distance δ from the robot.

This parameter depends on the BaseSpeed and on the Reaction Time parameter α, so it is dynamic and it is central for determining the safety property (we will discuss their proper value later).
The CloseTo signals raise the ChangeSpeed(operatingSpeed,down) event in order to avoid dangerous contacts. The all possible CloseTo signals show the different reaction of the Robot depending on

We have grouped specific signals into more generic signals to reduce repetition of axioms when possible.

k68.
$$CloseTo \Leftrightarrow BaseCloseTo \vee EECloseTo \vee ArmBaseCloseTo$$

k69.
$$BaseCloseTo \Leftrightarrow BaseCloseToBody \vee BaseCloseToArms \vee BaseCloseToHead$$

k70.
$$EECloseTo \Leftrightarrow EECloseToBody \vee EECloseToArms \vee EECloseToHead$$

k71.
$$ArmBaseCloseTo \Leftrightarrow ArmBaseCloseToBody \vee ArmBaseCloseToArms \vee ArmBaseCloseToHead$$

If the robot has not been halted by the operator (with the HDI) and there are no CloseTo signals the Robot tries to move/work with the fastest speed possible.
NOTE: the ChangeSpeed for the speed up is raised only if there was not other ChangeSpeed signals (up or down) for the previous α instants. In this way there is no possibility to repete the same signal for changing a certain speed and/or to speed up while it is slowing down.

k72.
$$Alw(\forall operatingSpeed(\neg RobotInactive \wedge \neg CloseTo \wedge \neg BaseSpeed(high) \wedge \forall accelerationDirection(Lasted_{ee}(\neg ChangeSpeed(operatingSpeed, accelerationDirection), \alpha) \Rightarrow ChangeSpeed(operatingSpeed, up)))$$

BaseCloseToBody

k73.
$$Alw(\exists cell_i, cell_j(UpToNow(\neg BaseCloseToBody) \wedge HumanBodyPosition(cell_i) \wedge BasePosition(cell_j) \wedge$$

$$\exists v((v \leq \delta_B) \wedge Distance(cell_i, cell_j, v))) \Rightarrow Becomes(BaseCloseToBody))$$

k74.

$$Alw(\ \exists cell_i, cell_j(UpToNow(BaseCloseToBody) \wedge$$
$$HumanBodyPosition(cell_i) \wedge BasePosition(cell_j) \wedge$$
$$\exists v((v > \delta_B) \wedge Distance(cell_i, cell_j, v)) \Rightarrow Becomes(\neg BaseCloseToBody))$$

When BaseCloseToBody is true, the robot slows down until its speed is "still" (obviously if the robot is already "still", the signal does not arise).

N.B.: ChangeSpeed is an event, so we must ensure that the robot receives this command every α instants, otherwise there could be a repetition of the same signal.
NOTE: decelleration, differently from the acceleration, can interrupt an acceleration ChangeSpeed avoiding the dangerous case in which the Robot speeds up while it is already active a CloseTo signal.

k75.

$$\forall operatingSpeed(BaseCloseToBody \wedge Lasted_{ee}(\neg ChangeSpeed(operatingSpeed, down), \alpha)) \wedge$$
$$\neg BaseSpeed(still) \Rightarrow ChangeSpeed(operatingSpeed, down)$$

BaseCloseToArms

k76.

$$Alw(\ \exists cell_i, cell_j, cell_k, cell_z(UpToNow(\neg BaseCloseToArms) \wedge$$
$$HumanArmPosition(cell_i, cell_j, cell_k) \wedge BasePosition(cell_z) \wedge$$
$$\exists v((v \leq \delta_B) \wedge (Distance(cell_i, cell_z, v) \vee Distance(cell_j, cell_z, v) \vee Distance(cell_k, cell_z, v)))) \Rightarrow$$
$$Becomes(BaseCloseToArms))$$

k77.

$$Alw(\ \exists cell_i, cell_j, cell_j, cell_z(UpToNow(BaseCloseToArms) \wedge$$
$$HumanArmPosition(cell_i) \wedge BasePosition(cell_j) \wedge$$
$$\exists v((v > \delta_B) \wedge (Distance(cell_i, cell_z, v) \vee Distance(cell_j, cell_z, v) \vee Distance(cell_k, cell_z, v)))) \Rightarrow$$
$$Becomes(\neg BaseCloseToArms))$$

When BaseCloseToArms is true, the robot slows down until its speed is "still"

k78.

$$\forall operatingSpeed(BaseCloseToArms \wedge Lasted_{ee}(\neg ChangeSpeed(operatingSpeed, down), \alpha)) \wedge$$
$$\neg BaseSpeed(still) \Rightarrow ChangeSpeed(operatingSpeed, down)$$

BaseCloseToHead

k79.

$$Alw(\ \exists cell_i, cell_j(UpToNow(\neg BaseCloseToHead) \wedge HumanHeadPosition(cell_i) \wedge BasePosition(cell_j) \wedge \exists v((v \leq \delta_B) \wedge Distance(cell_i, cell_j, v))) \Rightarrow Becomes(BaseCloseToHead))$$

k80.

$$Alw(\exists cell_i, cell_j(UpToNow(BaseCloseToHead) \wedge HumanHeadPosition(cell_i) \wedge BasePosition(cell_j) \wedge \exists v((v > \delta_B) \wedge Distance(cell_i, cell_j, v)) \Rightarrow Becomes(\neg BaseCloseToHead))$$

When BaseCloseToHead is true, the robot slows down until its speed is “still”

k81.

$$\forall operatingSpeed(BaseCloseToHead \wedge Lasted_{ee}(\neg ChangeSpeed(operatingSpeed, down), \alpha)) \wedge \neg BaseSpeed(still) \Rightarrow ChangeSpeed(operatingSpeed, down)$$

EECloseToBody

k82.

$$Alw(\ \exists cell_i, cell_j(UpToNow(\neg EECloseToBody) \wedge HumanBodyPosition(cell_i) \wedge EEPosition(cell_j) \wedge \exists v((v \leq \delta_E) \wedge Distance(cell_i, cell_j, v))) \Rightarrow Becomes(EECloseToBody))$$

k83.

$$Alw(\exists cell_i, cell_j(UpToNow(EECloseToBody) \wedge HumanBodyPosition(cell_i) \wedge EEPosition(cell_j) \wedge \exists v((v > \delta_E) \wedge Distance(cell_i, cell_j, v)) \Rightarrow Becomes(\neg EECloseToBody))$$

EECloseToArms

k84.

$$Alw(\ \exists cell_i, cell_j, cell_j, cell_z(UpToNow(\neg EECloseToArms) \wedge HumanArmPosition(cell_i) \wedge EEPosition(cell_j) \wedge \exists v((v \leq \delta_E) \wedge (Distance(cell_i, cell_z, v) \vee Distance(cell_j, cell_z, v) \vee Distance(cell_k, cell_z, v)))) \Rightarrow$$

$Becomes(EECloseToArms))$

k85.

$Alw(\exists cell_i, cell_j, cell_j, cell_z(UpToNow(EECloseToArms) \wedge$
$HumanArmPosition(cell_i) \wedge EEPosition(cell_j) \wedge$
$\exists v((v > \delta_E) \wedge (Distance(cell_i, cell_z, v) \vee Distance(cell_j, cell_z, v) \vee Distance(cell_k, cell_z, v)))) \Rightarrow$
$Becomes(\neg EECloseToArms))$

EECloseToHead

k86.

$Alw(\ \exists cell_i, cell_j(UpToNow(\neg EECloseToHead) \wedge$
$HumanHeadPosition(cell_i) \wedge EEPosition(cell_j) \wedge$
$\exists v((v \leq \delta_E) \wedge Distance(cell_i, cell_j, v))) \Rightarrow Becomes(EECloseToHead))$

k87.

$Alw(\exists cell_i, cell_j(UpToNow(EECloseToHead) \wedge$
$HumanHeadPosition(cell_i) \wedge EEPosition(cell_j) \wedge$
$\exists v((v > \delta_E) \wedge Distance(cell_i, cell_j, v)) \Rightarrow Becomes(\neg EECloseToHead))$

ArmBaseCloseToBody

k88.

$Alw(\ \exists cell_i, cell_j(UpToNow(\neg ArmBaseCloseToBody) \wedge$
$HumanBodyPosition(cell_i) \wedge ArmBasePosition(cell_j) \wedge$
$\exists v((v \leq \delta_E) \wedge Distance(cell_i, cell_j, v))) \Rightarrow Becomes(ArmBaseCloseToBody))$

k89.

$Alw(\exists cell_i, cell_j(UpToNow(ArmBaseCloseToBody) \wedge$
$HumanBodyPosition(cell_i) \wedge ArmBasePosition(cell_j) \wedge$
$\exists v((v > \delta_E) \wedge Distance(cell_i, cell_j, v)) \Rightarrow Becomes(\neg ArmBaseCloseToBody))$

When EECloseToBody or ArmBaseCloseToBody are true, the robot slows down until its speed is “slow”.

k90.

$\forall operatingSpeed((ArmBaseCloseToBody \vee EECloseToBody) \wedge (EESpeed(high) \vee EESpeed(medium)) \wedge$
$Lasted_{ee}(\neg ChangeSpeed(operatingSpeed, down), \alpha) \Rightarrow ChangeSpeed(operatingSpeed, down)$

ArmBaseCloseToArms

k91.

$$Alw(\exists cell_i, cell_j, cell_j, cell_z(UpToNow(\neg ArmBaseCloseToArms) \wedge HumanArmPosition(cell_i) \wedge ArmBasePosition(cell_j) \wedge \exists v((v \leq \delta_E) \wedge (Distance(cell_i, cell_z, v) \vee Distance(cell_j, cell_z, v) \vee Distance(cell_k, cell_z, v)))) \Rightarrow Becomes(ArmBaseCloseToArms))$$

k92.

$$Alw(\exists cell_i, cell_j, cell_j, cell_z(UpToNow(ArmBaseCloseToArms) \wedge ArmBaseCloseToArms) \wedge HumanArmPosition(cell_i) \wedge ArmBasePosition(cell_j) \wedge \exists v((v \leq \delta_E) \wedge (Distance(cell_i, cell_z, v) \vee Distance(cell_j, cell_z, v) \vee Distance(cell_k, cell_z, v)))) \Rightarrow Becomes(\neg ArmBaseCloseToArms))$$

When EECloseToArms or ArmBseCloseToArms are true, the robot slows down until its speed is “slow”.

k93.

$$\forall operatingSpeed((ArmBaseCloseToArms \vee EECloseToArms) \wedge (EESpeed(high) \vee EESpeed(medium)) \wedge Lasted_{ee}(\neg ChangeSpeed(operatingSpeed, down), \alpha) \Rightarrow ChangeSpeed(operatingSpeed, down)$$

ArmBaseCloseToHead

k94.

$$Alw(\exists cell_i, cell_j(UpToNow(\neg ArmBaseCloseToHead) \wedge HumanHeadPosition(cell_i) \wedge ArmBasePosition(cell_j) \wedge \exists v((v \leq \delta_E) \wedge Distance(cell_i, cell_j, v))) \Rightarrow Becomes(ArmBaseCloseToHead))$$

k95.

$$Alw(\exists cell_i, cell_j(UpToNow(ArmBaseCloseToHead) \wedge$$

$ArmBaseCloseToHead) \wedge HumanHeadPosition(cell_i) \wedge ArmBasePosition(cell_j) \wedge$
$\exists v((v \geq \delta_E) \wedge Distance(cell_i, cell_j, v)) \Rightarrow Becomes(\neg ArmBaseCloseToHead))$

When EECloseToHead or ArmBaseCloseToHead is true, the robot slows down until its speed is "still".

k96.
$\forall operatingSpeed((ArmBaseCloseToHead \vee EECloseToHead) \wedge \neg EESpeed(still) \wedge$
$Lasted_{ee}(\neg ChangeSpeed(operatingSpeed, down), \alpha) \Rightarrow ChangeSpeed(operatingSpeed, down)$

Robot does not entrap the operator

These axioms model the behavior of the robot when it is near the operator and near the wall, the robot has to avoid the situation in which the operator is entrapped between the robot and the wall.
If the operator is between the robot and the wall and the distance of the robot from the wall is less than 8 cells (and so there are 7 cells between the center of the robot and the wall) then the robot should not decrease its distance from the operator (so it must stop or move farther away from the operator).

$d_{wo}, d_{rw}, d_{ro1}, d_{ro2}$ are, respectively:

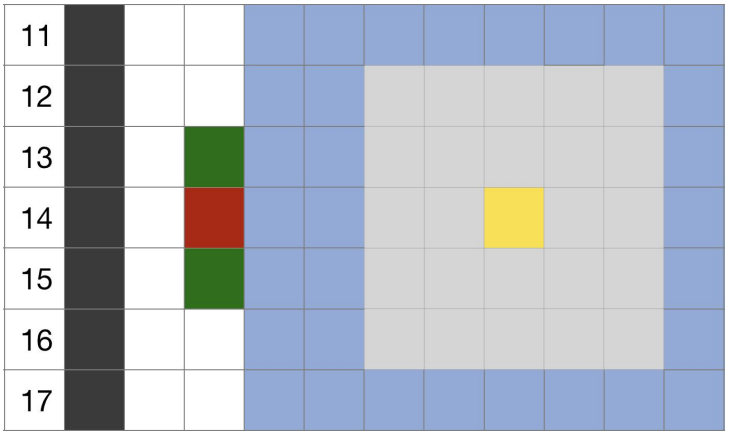
- The distance between the wall and the operator
- The initial distance between the robot and the wall
- The initial distance between the robot and the operator
- The succeeding distance between the robot and the operator in the case in which the robot moves in order to move away from the operator.

$cell_{wall}, cell_{op1}, cell_{op2}, cell_{r1}, cell_{r2}$ are, respectively:

- A cell of the wall
- The two cells in which the operator is located (before and after)
- The two cells in which the robot is located (before and after).

k97.
$\forall cell_{wall}(Wall(cell_{wall}) \wedge \exists cell_{op1},\ cell_{op2},\ cell_{r1},\ cell_{r2},\ d_{wo},\ d_{wr},\ d_{ro1},\ d_{ro2}($
$UpToNow($
$HumanBodyPosition(cell_{op1}) \wedge Distance(cell_{op1},\ cell_{wall},\ d_{wo}) \wedge$
$BasePosition(cell_{r1}) \wedge Distance(cell_{r1}, cell_{op1}, d_{ro}) \wedge Distance(cell_{r1}, cell_{wall}, d_{wr}) \wedge$
$d_{wo} < d_{wr} \wedge d_{wr} \leq 6\ ) \Rightarrow$
$NowOn(\ BasePosition(cell_{r1}) \vee$
$(BasePosition(cell_{r2}) \wedge\ HumanBodyPosition(cell_{i2})\ \wedge$
$Distance(cell_{k2}, cell_{j2}, d_4)\ \wedge\ d_4 > d_3\ )))$

What this is saying:
If the operator is closer to a particular cell of the wall than the robot and the robot is less than 8 cells away from that wall; in the next instant the robot is either still or it is moving but the distance robot-operator has increased.



Here the wall cell considered is (0, 14).
Operator distance from wall is 2
Base distance from wall is 7

Robot Workflow

Destination of the robot in mobile mode

At every instant, in absence of signals, the robot is either loading/unloading WorkPieces or it is moving between the LoadArea and the ReleaseArea.
These axioms model the destination of the robot (when it is moving in mobile mode), relying of the last area in which the robot has worked (ReleaseArea or LoadArea).

If the robot is in mobile mode and its last position in a working area was in the LoadArea then it is moving to the ReleaseArea without returning in the LoadArea in the meantime; the robot's bin has to be not empty too, otherwise it has no sense to move to the ReleaseArea.

k98.

$$UpToNow(Mode(mobile) \wedge \neg BinEmpty) \wedge \exists d_1(Past(BasePosition(cell_j) \wedge LoadArea(cell_j),\ d_1) \wedge Lasted_{ii}(\neg(BasePosition(cell_i) \wedge ReleaseArea(cell_i)), d1)) \Rightarrow \exists d_2(Futr(BasePosition(cell_k) \wedge ReleaseArea(cell_k),\ d_2) \wedge Lasts_{ii}(\neg(BasePosition(cell_z) \wedge LoadArea(cel_z)), d_2))$$

This is the vice versa of the previous axiom.

k99.

$$UpToNow(Mode(mobile) \wedge \neg BinFull) \wedge \exists d_1(Past(BasePosition(cell_j) \wedge ReleaseArea(cell_j),\ d_1) \wedge Lasted_{ii}(\neg(BasePosition(cell_i) \wedge LoadArea(cell_i)), d1)) \Rightarrow \exists d_2(Futr(BasePosition(cell_k) \wedge LoadArea(cell_k),\ d_2) \wedge Lasts_{ii}(\neg(BasePosition(cell_z) \wedge ReleaseArea(cel_z)), d_2))$$

Working in Manipulator Mode

If the Robot is "working on an operation" (i.e. it is operating at a workstation) then it is operating in "manipulator" mode.

k100.

$$Alw(\exists operation\ (Working(operation)) \Rightarrow Mode(manipulator))$$

Working(pickWP)

The "Working(operation)" predicate states that, when the robot reaches the BinArea and switches to "manipulator" Mode, it will keep operating at that work station until the bin is full or it receives an interrupt signal.

k101.

$$Since(BaseSpeed(Still) \wedge \neg BinFull \wedge \neg Signal \wedge \neg RobotInactive, BasePosition(cell_i) \wedge LoadArea(cell_i)) \Leftrightarrow Working(pickWP)$$

k102.

$$Working(pickWP) \Rightarrow CollectingFromBin \vee PlacingInLocalBin \vee Moving$$

k103.

$$Moving \Rightarrow \neg ArmSpeed(still)$$

k104.

$$CollectingFromBin \Rightarrow \forall cell_i(EEPosition(cell_i) \Rightarrow BinArea(cell_i))$$

k105.

$$PlacingInLocalBin \Rightarrow \forall cell_i(EEPosition(cell_i) \Rightarrow BinPosition(cell_i))$$

The two following axioms ensure the alternating of the actions.

k106.

$$PlacingInLocalBin \Rightarrow Since(\neg PlacingInLocalBin, CollectingFromBin \wedge Working(pickWP))$$

k107.

$$CollectingFromBin \Rightarrow Since(\neg CollectingFromBin, PlacingInLocalBin \wedge Working(pickWP))$$

Working(putWP)

The "Working(operation)" predicate states that, when the robot reaches the WorkArea and switches to "manipulator" Mode, it will keep operating at that work station until the bin is full or it receives an interrupt signal.

k108.

$$Since(BaseSpeed(still) \wedge \neg BinEmpty \wedge \neg Signal, BasePosition(cell_i) \wedge ReleaseArea(cell_i)) \Leftrightarrow Working(putWP)$$

k109.

$$Working(putWP) \Rightarrow CollectingFromLocalBin \vee PlacingInWorkArea \vee Moving$$

k110.

$$Moving \Rightarrow \neg ArmSpeed(still)$$

k111.
$$CollectingFromLocalBin \Rightarrow \forall cell_i(EEPosition(cell_i) \Rightarrow binPosition(cell_i))$$

k112.
$$PlacingInWorkArea \Rightarrow \forall cell_i(EEPosition(cell_i) \Rightarrow workArea(cell_i))$$

The two following axioms ensure the alternating of the actions.

k113.
$$PlacingInWorkArea \Rightarrow Since(\neg PlacingInWorkingArea, CollectingFromLocalBin \wedge Working(putWP))$$

k114.
$$CollectingFromLocalBin \Rightarrow Since(\neg CollectingFromLocalBin, PlacingInWorkingArea \wedge Working(putWP))$$

Ending PickWP/PutWP

When the robot has completed its task at a working station, it will move its arm in the rest postiition

k115.
$$UpToNow(Working(pickWP)) \wedge Becomes(BinFull)) \Rightarrow SomF(ArmInRestPosition)$$

k116.
$$UpToNow(Working(putWP)) \wedge Becomes(BinEmpty)) \Rightarrow SomF(ArmInRestPosition)$$

Safety Properties

The First Safety Property requires that the Robot's Base does not come in contact with the operator while moving (if any contact occurs, BaseSpeed must be "still").

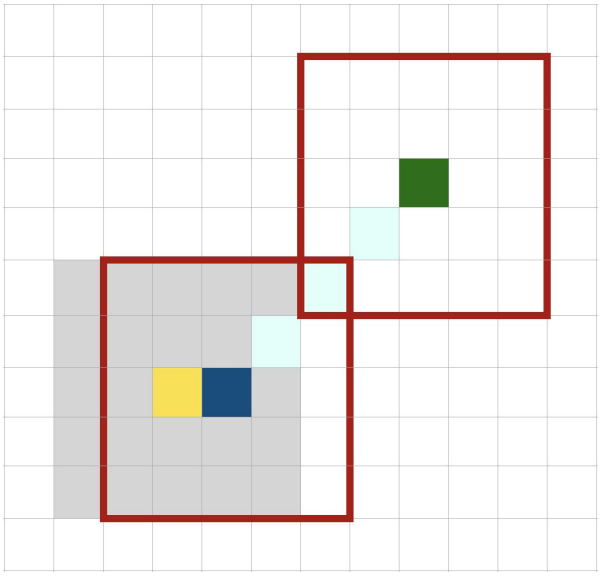
S1. $$\exists cell_i, cell_j, d\ (HumanBodyPosition(cell_i) \wedge BasePostion(cell_j) \wedge Distance(cell_i, cell_j, d) \wedge d \leq 3) \Rightarrow BaseSpeed(still)$$

S2. $$\exists cell_i, cell_j, cell_k, cell_z, d\ (HumanArmPosition(cell_i, cell_j, cell_k) \wedge BasePostion(cell_z) \wedge (Distance(cell_i, cell_z, d) \vee Distance(cell_j, cell_z, d) \vee Distance(cell_k, cell_z, d)) \wedge d \leq 3) \Rightarrow$$

$BaseSpeed(still)$

S3. $\exists\, cell_i, cell_j, d\ (HumanHeadPosition(cell_i) \wedge BasePostion(cell_j) \wedge$
$Distance(cell_i, cell_j, d) \wedge d \leq 3) \Rightarrow$
$BaseSpeed(still)$

The Second Safety Property is about the EESpeed and is more complex. Actually we want to ensure that if the body or the arms of the operator are close to the Robot Arm, the latter can stay still or continue to move but only with slow speed. This is because the operator must be able to relocate the robot's arm.
On the other hand, if the Human Head is too close, the Robot Arm must be still.
The secure area that we want to create around the Robot Arm is composed by a 5x5 square of cells around the EE and another 5x5 square of cells around the Arm Base.



The two red perimeters delimit the danger area around the ArmBase and the EndEffector, ensuring that the whole arm is covered.

S4. $\exists\, cell_i,\ cell_j,\ cell_k,\ d\ \ (HumanBodyPosition(cell_i) \wedge EEPostion(cell_j) \wedge$
$ArmBasePosition(cell_k) \wedge d < 2 \wedge$
$(Distance(cell_i, cell_j, d) \vee Distance(cell_i, cell_k, d))) \Rightarrow$
$EESpeed(low) \vee EESpeed(still)$

S5. $\exists\, cell_i, cell_j, cell_k, cell_w, cell_z, d\ (HumanArmsPosition(cell_i, cell_j, cell_k) \wedge EEPostion(cell_w) \wedge$
$ArmBasePosition(cell_z) \wedge d \leq 2 \wedge$
$(Distance(cell_i, cell_w, d) \vee Distance(cell_j, cell_w, d) \vee Distance(cell_k, cell_w, d) \vee$
$Distance(cell_i, cell_z, d) \vee Distance(cell_j, cell_z, d) \vee Distance(cell_k, cell_z, d))) \Rightarrow$
$EESpeed(low) \vee EESpeed(still)$

S6. $\exists\, cell_i, cell_j, cell_k, d_1, d_2 (HumanHeadPosition(cell_i) \wedge EEPostion(cell_j) \wedge$
$ArmBasePosition(cell_k) \wedge (d_1 \leq 2 \wedge Distance(cell_i, cell_j, d_1)) \vee$
$(d_2 \leq 2 \wedge Distance(cell_i, cell_k, d_2)) \Rightarrow EESpeed(still)$

In order to achieve these fundamental properties in our model we have to deal with the parametric values of α *and* δ. With the former we characterized the way in which the Robot can change its speed, a sort of Reaction Time parameter. Thanks to α we are then able to define the δ_{min} value that determines the minimum distance at which the specific CloseTo signal must arise in order to imply correctly the Safety Property (α is constant for every speed change).
First of all, it has to be greater than the cells which the Robot will pass through before stopping:

$$\delta_{min} \geq (3 \cdot \nu_h + 2 \cdot \nu_m + 1 \cdot \nu_l) \cdot \alpha + k \qquad \nu_h, \nu_m, \nu_l \in \{0, 1\},\ k \in \{0,\ 1,\ 2\}$$

Where ν_h, ν_m, ν_l are equal to 1 if in the process of slowing down, the robot passes through the relative speed (respectively: high, medium, low).

The parameter k has been added because, since our time is discrete, in one time instant the robot can surpass our boundary by more than one cell. If at one time instant, the BasePosition is next to the boundary, and the robot is moving towards the operator, on the next time instant the robot will be 1, 2 or 3 cells inside the boundary (depending on its speed). So k is equal to 0, 1, 2 respectively for when the robot is travelling at low, medium, high speed.

A few examples:

$BaseSpeed(high) \rightarrow BaseSpeed(still) \quad \Rightarrow [\nu_h, \nu_m, \nu_l] = [1,\ 1,\ 1]$
$EESpeed(high) \rightarrow EESpeed(still) \quad \Rightarrow [\nu_h, \nu_m, \nu_l] = [1,\ 1,\ 1]$
$BaseSpeed(high) \rightarrow BaseSpeed(medium) \quad \Rightarrow [\nu_h, \nu_m, \nu_l] = [1,\ 0,\ 0]$
$EESpeed(medium) \rightarrow EESpeed(low) \quad \Rightarrow [\nu_h, \nu_m, \nu_l] = [0,\ 1,\ 0]$
etc...

So, based on the initial speed, the robot will be still after:

- 3α time instants if the initial speed is high, and in this time it will go through $(3 \cdot 1 + 2 \cdot 1 + 1 \cdot 1) \cdot \alpha + k = 6\alpha + 2$cells.
- 2α time instants if the initial speed is medium, and in this time it will go through $(3 \cdot 0 + 2 \cdot 1 + 1 \cdot 1) \cdot \alpha + k = 3\alpha + 1$ cells.
- 1α time instants if the initial speed is low, and in this time it will go through $(3 \cdot 0 + 2 \cdot 0 + 1 \cdot 1) \cdot \alpha = 1\alpha$ cells.
- 0 time instants if the initial speed is already still

Now, we have to adjust the formula to deal with the RobotArm and the RobotBase separately.

For the RobotBase, it is enough to add +3 (radius of base plus 1) to the formula in order to get the right distance to avoid contact with the Operator.

For the Robot Arm, we take into account two different areas (as shown before): the EE danger area and the ArmBase one, both with a radius of 2 (thus we add 3 for them too).
Furthermore we add $(\nu_h + \nu_m)$ in oreder to be sure

Base braking $\delta_{min} \geq (3 \cdot \nu_h + 2 \cdot \nu_m + 1 \cdot \nu_l) \cdot \alpha + k + 3$
EE/ArmBase braking $\delta_{min} \geq (3 \cdot \nu_h + 2 \cdot \nu_m + 1 \cdot \nu_l) \cdot \alpha + k + 3$

Finally we must consider the Operator speed.
We assume the worst case scenario: we denote with μ_{MAX} the maximum number of cells that a human can cover in one time instant.

Cells covered in one time instant:	μ_{MAX}
Time instants elapsed during robot slow-down:	$(\nu_h + \nu_m + \nu_l) \cdot \alpha$

So the total distance covered by the human in the time it takes the robot to stop is:

$$(\nu_h + \nu_m + \nu_l) \cdot \alpha \cdot \mu_{MAX}$$

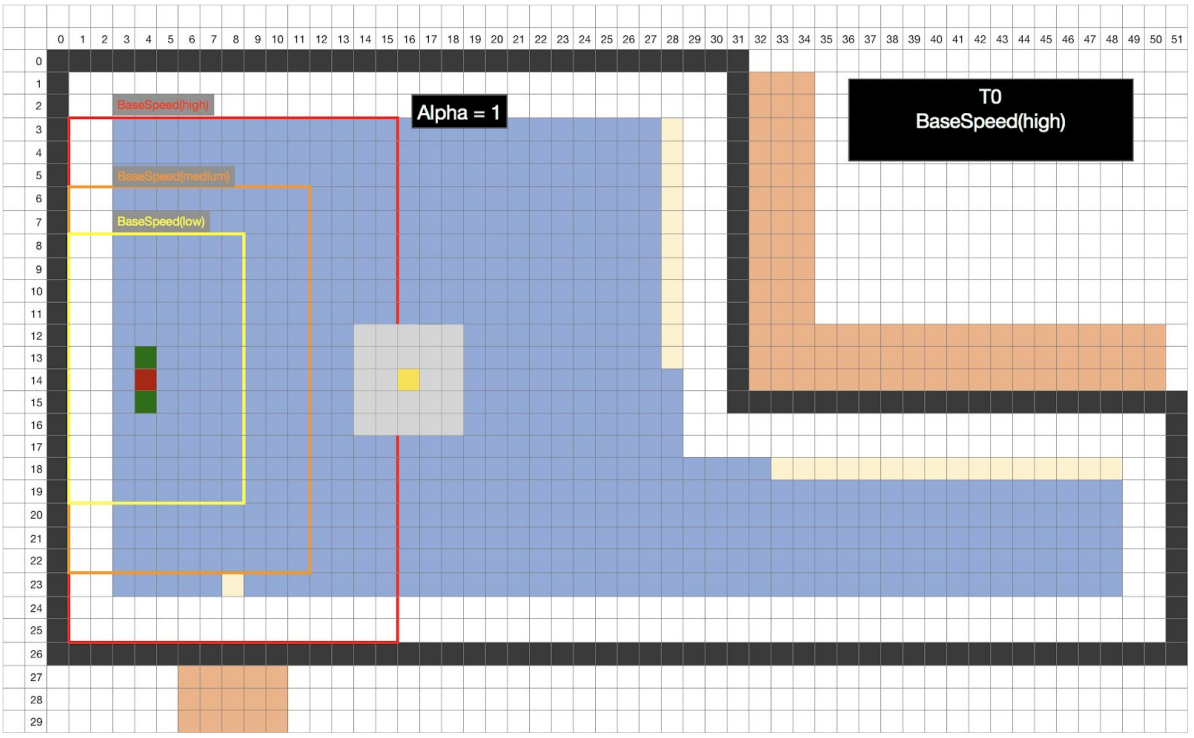
In conclusion, we can define δ as being greater or equal to δ_{min}, which highlights the minimum requirement for ensuring the Safety Property (we can decide to take a δ even bigger than the δ_{min} but this will affect the utility property):

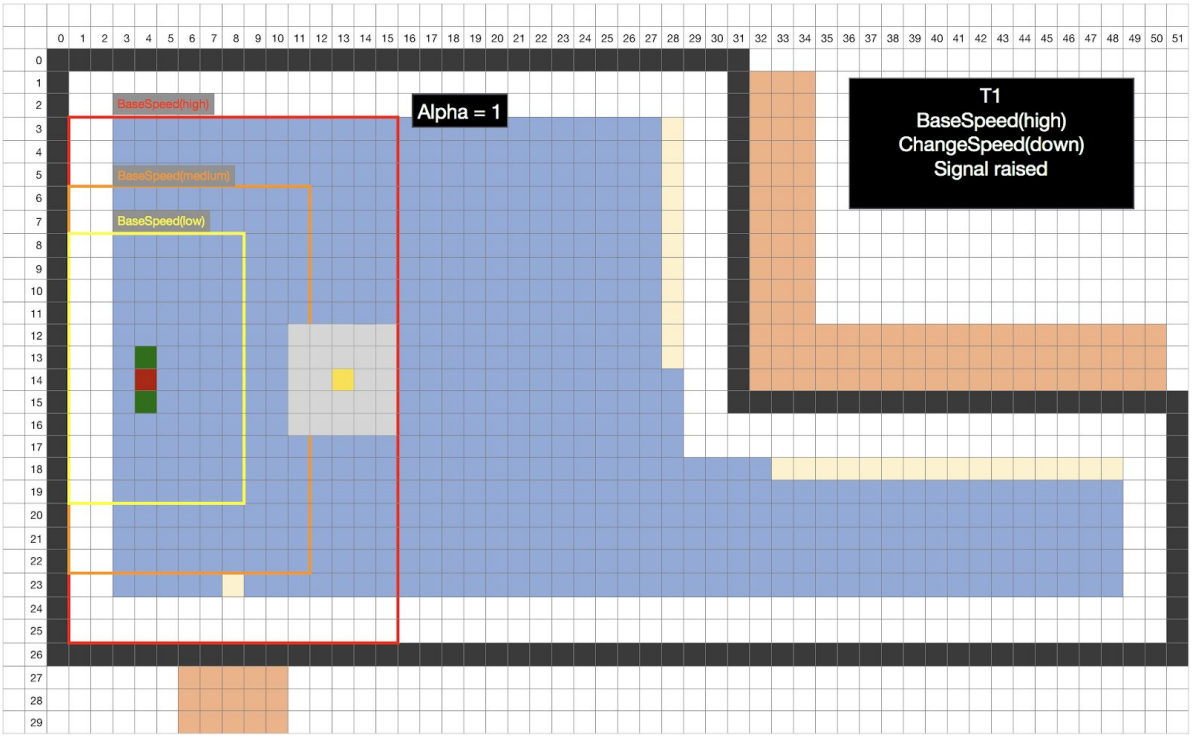
Base braking distance $\delta_B = (3 \cdot \nu_h + 2 \cdot \nu_m + 1 \cdot \nu_l) \cdot \alpha + k + 3 + (\nu_h + \nu_m + \nu_l) \cdot \alpha \cdot \mu_{MAX}$
EE braking distance $\delta_E = (3 \cdot \nu_h + 2 \cdot \nu_m + 1 \cdot \nu_l) \cdot \alpha + k + 3 + (\nu_h + \nu_m + \nu_l) \cdot \alpha \cdot \mu_{MAX}$

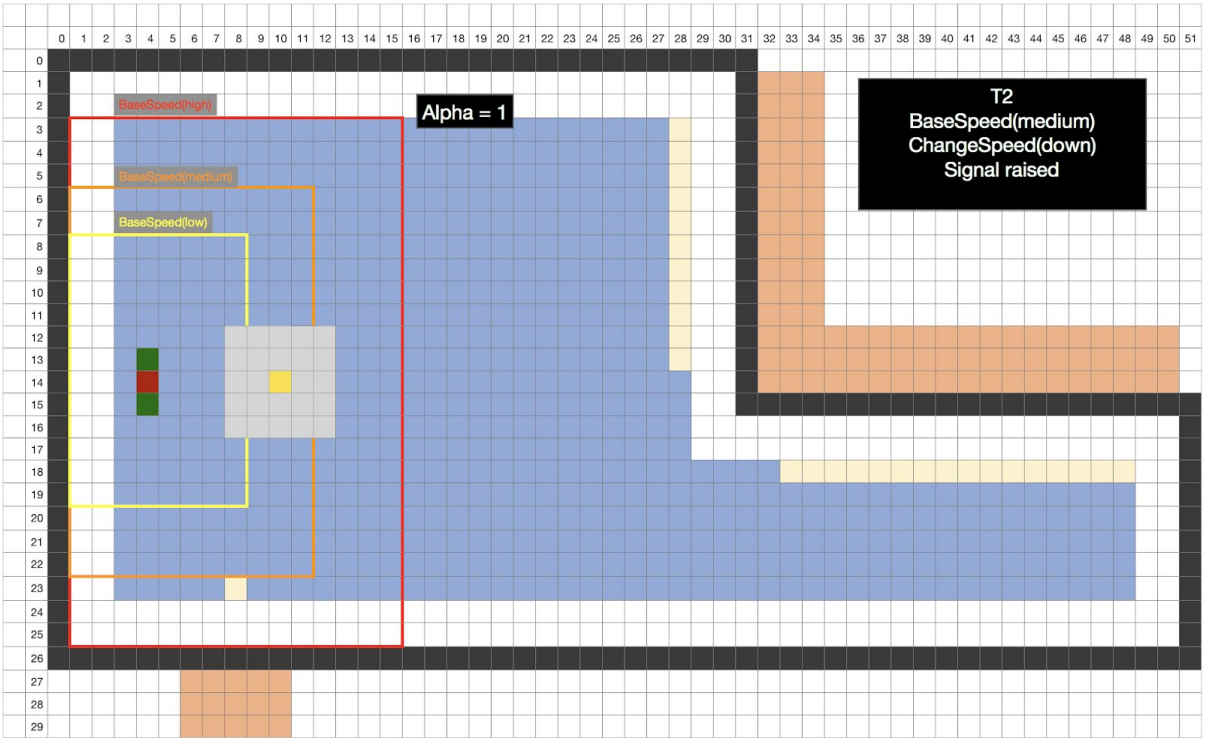
(finally we decide to use two different notations for δ even though they have the same value because they depend on two different speed: the Base's and the Arm's one)

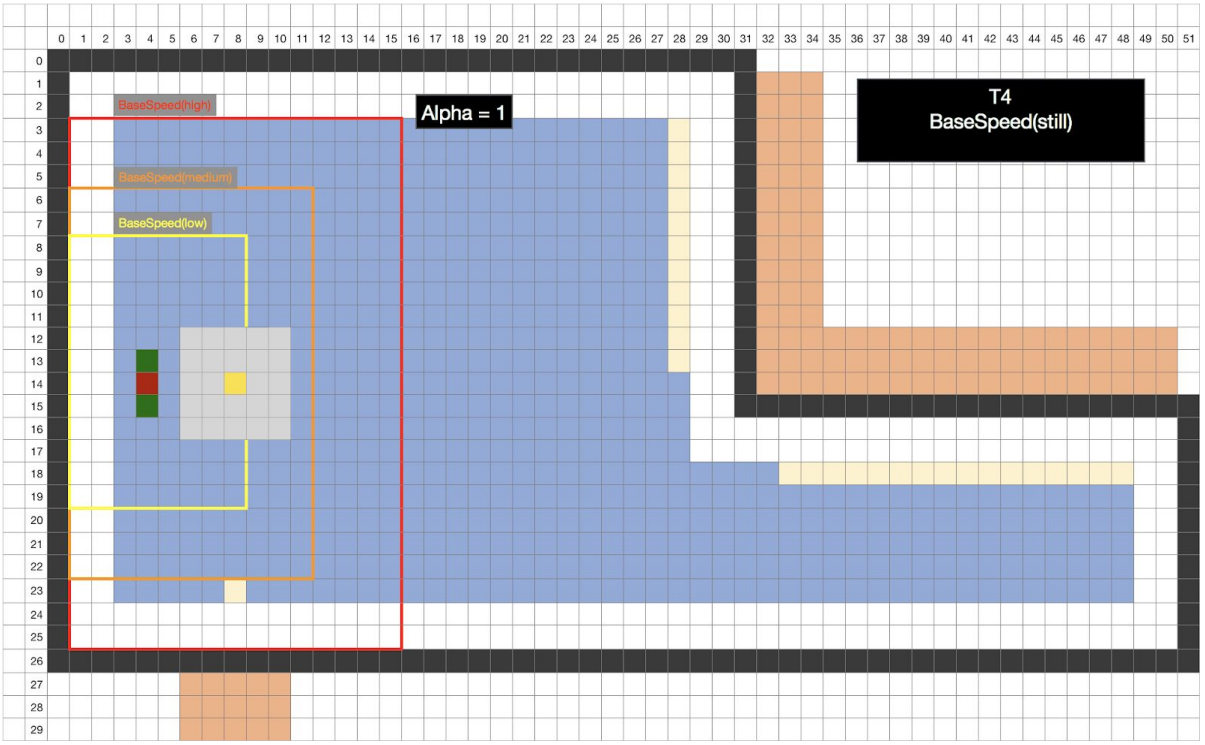
$$\delta \geq \delta_{min} = (3 \cdot \nu_h + 2 \cdot \nu_m + 1 \cdot \nu_l) \cdot \alpha + k + 3 + (\nu_h + \nu_m + \nu_l) \cdot \alpha \cdot \mu_{MAX}$$

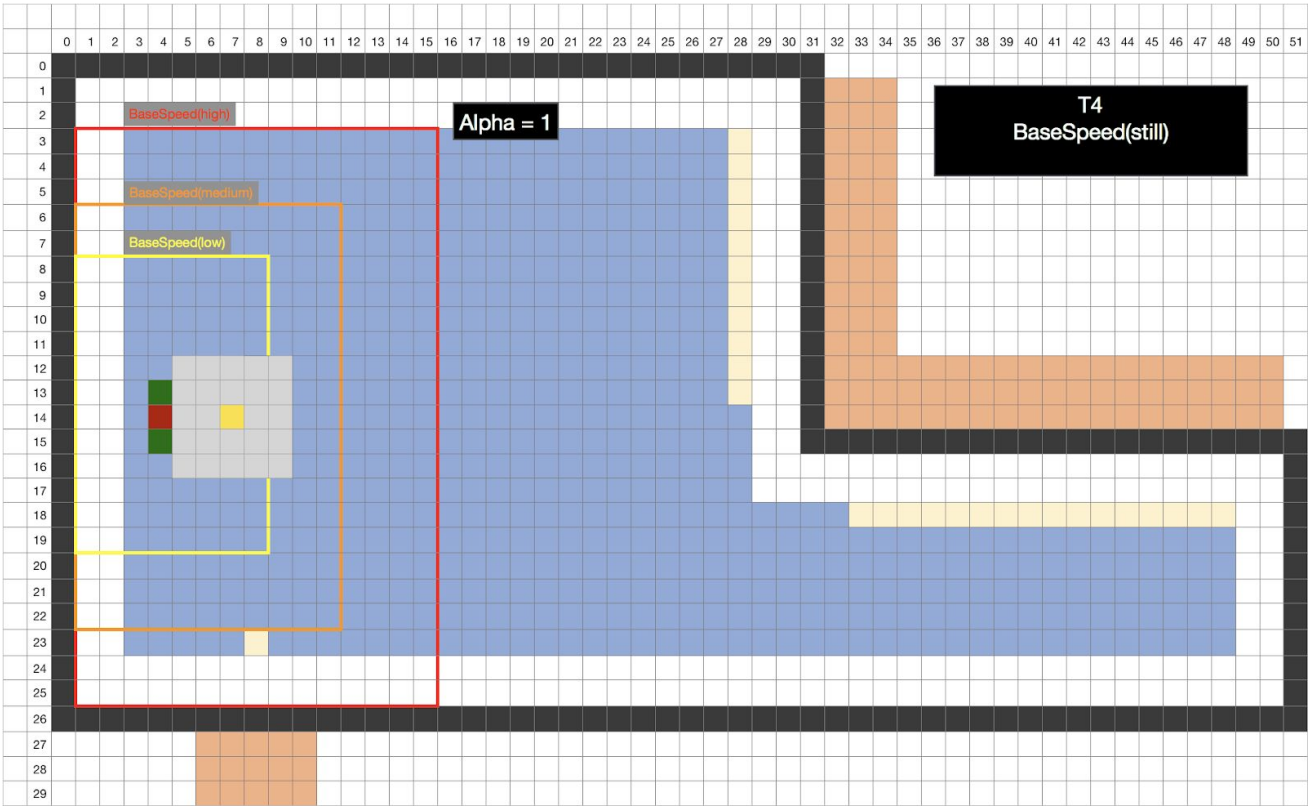
Here is a simplified example, where we consider the operator not to move ($\mu_{MAX} = 0$) and the reaction time of the robot is equal to 1 ($\alpha = 1$). The colored lines around the operator delimit the danger area according to the speed of the robot.


BaseSpeed(high)
BaseSpeed(medium)
BaseSpeed(low)
Alpha = 1
T0
BaseSpeed(high)


BaseSpeed(high)
BaseSpeed(medium)
BaseSpeed(low)
Alpha = 1
T1
BaseSpeed(high)
ChangeSpeed(down)
Signal raised


BaseSpeed(high)
BaseSpeed(medium)
BaseSpeed(low)
Alpha = 1
T2
BaseSpeed(medium)
ChangeSpeed(down)
Signal raised


BaseSpeed(high)
BaseSpeed(medium)
BaseSpeed(low)
Alpha = 1
T4
BaseSpeed(still)



The robot gradually slows down, reaching a full stop before hitting the operator.

Discussion about the safety properties in the model

Here we want to prove that our model of the system assures that the safety properties are never violated.

➢ S1: if the distance between the Robot's center and the operator is ≤ 3 then the Robot's base is still.

The robot can be only in two modes: manipulator or mobile.

(k28.) $Alw(Mode(mobile) \Leftrightarrow \neg Mode(manipulator))$

If the robot is in manipulator mode then its speed is already "still" and so S1 is satisfied:

(k30.) $Alw(Mode(manipulator) \Rightarrow BaseSpeed(still))$

Otherwise, if the mode is mobile, when the operator reach a distance of δ_B from the robot's base, the signal BaseCloseToBody will be activated,

(k73.) $Alw(\exists cell_i, cell_j(UpToNow(\neg BaseCloseToBody) \wedge$
$HumanBodyPosition(cell_i) \wedge BasePosition(cell_j) \wedge$
$\exists v((v \leq \delta_B) \wedge Distance(cell_i, cell_j, v))) \Rightarrow Becomes(BaseCloseToBody))$

at that point the Robot will receive the command of slowing down its speed.

(k75.) $\forall operatingSpeed(BaseCloseToBody \wedge Lasted_{ee}(\neg ChangeSpeed(operatingSpeed, down), \alpha)) \wedge$
$\neg BaseSpeed(still) \Rightarrow ChangeSpeed(operatingSpeed, down)$

In α time instants the robot will slow down from its speed to the lower one:

(k22.) $Mode(mobile) \wedge ChangeSpeed(high, down) \Rightarrow$
$Lasts_{ie}(BaseSpeed(high, \alpha) \wedge Futr(BaseSpeed(medium), \alpha)$

(k23.) $Mode(mobile) \wedge ChangeSpeed(medium, down) \Rightarrow$
$Lasts_{ie}(BaseSpeed(medium, \alpha) \wedge Futr(BaseSpeed(low, \alpha)$

(k24.) $Mode(mobile) \wedge ChangeSpeed(low, down) \Rightarrow$
$Lasts_{ie}(BaseSpeed(low, \alpha) \wedge Futr(BaseSpeed(still), \alpha)$

(k25.) $Mode(mobile) \wedge ChangeSpeed(still, down) \Rightarrow$
$Lasts_{ie}(BaseSpeed(still, \alpha) \wedge Futr(BaseSpeed(still), \alpha)$

and this is repeated until the Robot's speed becomes 'still'. (k75.)

At the end the robot will take $(\nu_h + \nu_m + \nu_l) \cdot \alpha$ time instants to stop and it will go through $(3 \cdot \nu_h + 2 \cdot \nu_m + 1 \cdot \nu_l) \cdot \alpha$ cells in this time; adding to this quantity the distance that, in this time, the operator can cover we arrive at the conclusion that the security distance between the operator and the robot, that triggers the signal BaseCloseToBody, has to be at least δ_B in order to assure S1.

Now, we analyze the scenario in which the operator gets closer than δ_B to the robot but then he moves away to a distance greater than δ_B. In this case the Robot will starts to decelerate but, in order to do not impact too much on the efficiency of the Robot, when the operator moves away at a distance greater than δ_B $BaseCloseToBody$ becomes false

(k73.) $Alw(\ \exists cell_i, cell_j(UpToNow(\neg BaseCloseToBody) \wedge$
$HumanBodyPosition(cell_i) \wedge BasePosition(cell_j) \wedge$
$\exists v((v \leq \delta_B) \wedge Distance(cell_i, cell_j, v))) \Rightarrow Becomes(BaseCloseToBody))$

and, if there aren't other 'CloseTo signals', also CloseTo becomes false.

(k68.) $CloseTo \Leftrightarrow BaseCloseTo \vee EECloseTo \vee ArmBaseCloseTo$
(k69.) $BaseCloseTo \Leftrightarrow BaseCloseToBody \vee BaseCloseToArms \vee BaseCloseToHead$

Thanks to this, there will be a signal of ChangeSpeed(up) that will let the robot accelerate again.

(k72.) $Alw(\forall operatingSpeed(\neg RobotInactive \wedge \neg CloseTo \wedge \neg BaseSpeed(high) \wedge$
$\forall accelerationDirection(Lasted_{ee}(\neg ChangeSpeed(operatingSpeed, accelerationDirection), \alpha) \Rightarrow$
$ChangeSpeed(operatingSpeed, up)))$

➢ S2: if the distance between the Robot's center and the operator's arm is ≤ 3 then the Robot's base is still.
This is assured by the model just like S1 but using the signal $BaseCloseToArms$ instead of $BaseCloseToBody$.

➢ S3: if the distance between the Robot's center and the operator's head is ≤ 3 then the Robot's base is still.
This is assured by the model just like S1 but using the signal $BaseCloseToHead$ instead of $BaseCloseToBody$.

➢ S4: if the distance between the operator's body and the End-effector or the arm's base is ≤ 2 then the speed of the arm is low or the arm is still.
As we have already explained in the demonstration of S1, the Robot can be only in two modes: manipulator or mobile.
If it is in mobile mode the arm has to be in RestPosition

(k29.) $Alw(Mode(mobile) \Rightarrow ArmInRestPosition)$

and this implies that the arm is inside the area occupied by the Robot's base and, in particular, that the arm is still

(k42.) $Alw(\ ArmInRestPosition \Leftrightarrow (ArmSpeed(still) \wedge \neg \exists cell_i,\ cell_j,\ cell_k($
$ArmPosition(cell_i,\ cell_j) \wedge BasePosition(cell_k) \wedge$
$(\exists v(v > 2 \wedge Distance(cell_k,\ cell_j,\ v)))$

On the other hand, if the Robot is in manipulator mode, when the operator's body reach a distance of δ_E from the Arm's base or from the End-effector, the signals, respectively, ArmBaseCloseToBody and EECloseToBody will be activated

(k88.) $Alw(\ \exists cell_i, cell_j(UpToNow(\neg ArmBaseCloseToBody) \wedge$
$HumanBodyPosition(cell_i) \wedge ArmBasePosition(cell_j) \wedge$
$\exists v((v \leq \delta_E) \wedge Distance(cell_i, cell_j, v))) \Rightarrow Becomes(ArmBaseCloseToBody))$

(k82.) $Alw(\ \exists cell_i, cell_j(UpToNow(\neg EECloseToBody) \wedge$
$HumanBodyPosition(cell_i) \wedge EEPosition(cell_j) \wedge$
$\exists v((v \leq \delta_E) \wedge Distance(cell_i, cell_j, v))) \Rightarrow Becomes(EECloseToBody))$

at that point the Robot will receive the command of slowing down its speed

(k90.) $\forall operatingSpeed((ArmBaseCloseToBody \vee EECloseToBody) \wedge (EESpeed(high) \vee EESpeed(medium)) \wedge$
$Lasted_{ee}(\neg ChangeSpeed(operatingSpeed, down), \alpha) \Rightarrow ChangeSpeed(operatingSpeed, down)$

and this, considering that the mode is manipulator, will bring to a slowdown of the ArmSpeed:

(k49.) $Mode(manipulator) \wedge ChangeSpeed(high, down) \Rightarrow$
$Lasts_{ie}(ArmSpeed(high, \alpha)) \wedge Futr(ArmSpeed(medium), \alpha)$

(k50.) $Mode(manipulator) \wedge ChangeSpeed(medium, down) \Rightarrow$
$Lasts_{ie}(ArmSpeed(medium, \alpha)) \wedge Futr(ArmSpeed(low), \alpha)$

(k51.) $Mode(manipulator) \wedge ChangeSpeed(low, down) \Rightarrow$
$Lasts_{ie}(ArmSpeed(low, \alpha)) \wedge Futr(ArmSpeed(still), \alpha)$

(k52.) $Mode(manipulator) \wedge ChangeSpeed(still, down) \Rightarrow$
$Lasts_{ie}(ArmSpeed(still, \alpha)) \wedge Futr(ArmSpeed(still), \alpha)$

and this is repeated until the Robot's speed becomes low or still.

Similarly to S1, the Robot's arm will take $(\nu_h + \nu_m + \nu_l) \cdot \alpha$ time instants to stop and to assure that the operator has not a distance ≤ 2 before the robot is at speed low (or still) it is necessary that the security distances from the operator's body to the End-effector and to the base of the arm, that trigger the signals EECloseToBody and ArmBaseCloseToBody, have to be at least δ_E in order to assure S4.

As in the previous cases, if the distances between the operator's body and the arm's parts (both End-effector and ArmBase) become again greater than δ_E then the signals (both or the only one that is active) become false

(k83.) $Alw(\exists cell_i, cell_j(UpToNow(EECloseToBody) \wedge$
$HumanBodyPosition(cell_i) \wedge EEPosition(cell_j) \wedge$
$\exists v((v > \delta_E) \wedge Distance(cell_i, cell_j, v)) \Rightarrow Becomes(\neg EECloseToBody))$

(k89.) $Alw(\exists cell_i, cell_j(UpToNow(ArmBaseCloseToBody) \wedge$
$HumanBodyPosition(cell_i) \wedge ArmBasePosition(cell_j) \wedge$
$\exists v((v > \delta_E) \wedge Distance(cell_i, cell_j, v)) \Rightarrow Becomes(\neg ArmBaseCloseToBody))$

and, if there aren't other 'CloseTo signals', also CloseTo becomes false.

(k68.) $CloseTo \Leftrightarrow BaseCloseTo \vee EECloseTo \vee ArmBaseCloseTo$

(k69.) $BaseCloseTo \Leftrightarrow BaseCloseToBody \vee BaseCloseToArms \vee BaseCloseToHead$

Thanks to this, there will be a signal of ChangeSpeed(up) and this is repeated until the arm's speed becomes high.

(k72.) $Alw(\forall operatingSpeed(\neg RobotInactive \wedge \neg CloseTo \wedge \neg BaseSpeed(high) \wedge$
$\forall accelerationDirection(Lasted_{ee}(\neg ChangeSpeed(operatingSpeed, accelerationDirection), \alpha) \Rightarrow$
$ChangeSpeed(operatingSpeed, up)))$

➢ S5: if the distance between the operator's arms and the End-effector or the arm's base is ≤ 2 then the speed of the arm is low or the arm is still.
This is assured by the model just like S4 but using the signal $EECloseToArms$ and $ArmBaseCloseToArms$ instead of $EECloseToBody$ and $ArmBaseCloseToBody$

➢ S6: if the distance between the operator's head and the End-effector or the arm's base is ≤ 2 then the speed of the arm is still.
This is assured by the model just like S4 but using the signal $EECloseToHead$ and $ArmBaseCloseToHead$ instead of $EECloseToBody$ and $ArmBaseCloseToBody$
And, differently from the case of the signals, the operation of slowdown is repeated until the arm's speed becomes still.

(k96.) $\forall operatingSpeed((ArmBaseCloseToHead \vee EECloseToHead) \wedge \neg EESpeed(still) \wedge$
$Lasted_{ee}(\neg ChangeSpeed(operatingSpeed, down), \alpha) \Rightarrow ChangeSpeed(operatingSpeed, down)$