# Methods and materials

## Manipulator arms

### Definition

Each manipulator arm configuration (configuration) is built from three parts: joint, joint axis and link length. The joints are divided into three types:

* Roll – Revolute joint about Z-axis
* Pitch – Revolute joint about Y-axis
* Prismatic – Linear sliding along Z-axis

The Joint axis is defined relative to the previous coordinate system, therefore according to the previous joint axis.

The link length is the length of each link along the Z-axis of its coordinate system.

in Figure 1 it can be seen an example for Roll\_Z\_0.1 🡪Pitch\_Y\_0.4 🡪 Prismatic\_Y\_0.7.

X

Y

Z

X

Y

Z

Y

X

Z

Figure 1- Configuration Example

### Parameters

For the creation of each configuration the following parameters have been set:

* All the links are cylindrical where the first link at radius 0.05m, the second 0.045m and continue until 0.035, 0.03, 0.025m for 4,5,6 DOF, respectively.
* For Revolute joints (Roll and Pitch) the joint limit is are [-PI, PI] and for prismatic joints, the limit is [0, 2\*link length]
* All the configuration will be between 3-6 Degrees of Freedom (DOF)
* The length of the links will be one of the following (0.1, 0.4, 0.7) meters
* Success: the arm need to reach to one of the 2 top points and to the middle and the lower points (Figure \*\*\*)

This gives about 34,636,800 theoretic configurations.

### Assumptions

In order to reduce the number of configurations and to fit the specific problem some assumptions have been made:

* The first joint is rotational along Z ax (Roll\_Z)
* The first link length = 0.1m
* The total length of all the links > 1m
* 2 adjacent prismatic joints must be perpendiculars
* No more than 3 prismatic joints
* After Roll joint can’t be Roll\Pitch joint in the Z-axis
* After Roll Joint or the following joints sequence [ Roll -> Pris Z] the next joint won’t be in the X-axis

After those assumptions, there are 1,695,044 configurations.

### Indices of manipulator performance

In order to make comparisons between the configurations and to be set as the optimization objectives several manipulators Indices and other indices have been checked:

All the indices that been checked are local indices, meaning indices that only depend on a specific position of the arm or the arm structure. Because the indices are calculated for several points the point with the lowest grade was selected as the grade of the configuration.

The chosen indices from the literature presented in figure 2.

Mid-Range Proximity Index doesn’t have an upper limit and in order to take into consideration the differences between the type of joints when calculating the index, we normalized the value by the joint limit, which is bounding the index between [0 – 0.5].



Figure 2- indices to examine

## Simulation

### Simulator

Gazebo simulator and MoveIt for motion planning, inverse kinematics, control, and collision checking, have been used In order to simulate different configurations of robotic arms. Robot Operating System (ROS) has been used to connect between those platforms and ease the process.

Gazebo simulates the real world and the environment where the selected manipulator arm will work. In the simulation, the plant is modeled as a cylinder at a height of 0.75 meters and a radius of 0.5 meters.

In order to control, each configuration movement rapidly exploring random tree (RRT) algorithm, which can handle problems with obstacles and differential constraints, selected as the path planner. RRT isn't finding the optimal path but find a path in a short time. In the simulation, the planner has a limit of 2 seconds to plan a path to the desired point.

The dynamic and kinematic structure that is used in ROS and Gazebo, to present arms, is Universal Robotics Description Format (URDF). Because the URDF also considers the dynamic part of the arm there is necessary to calculate the weight of each link. The weight of each link is calculated by the ratio between the accumulated length to the accumulated weight of the link- the ratio was calculated according to 2 different types of manipulators: UR5 and MOTOMAN NX100 (figure \*\*\*)

In the simulator, 4 points have been chosen to be reached by the arm. All the points are at a distance of 30 cm from the plant and in a different orientation.

\*\*\*\* Add the points + plant image with coordinate frame \*\*\*

Since the manipulator will be mounted on a mobile platform that will drive through the greenhouse, a prismatic joint, parallel to the plant for each configuration, was added in the base of the arm. The range of this prismatic joint is 1.5 meters so it will be able to move before in after the plant and the mobile platform position won’t be the reason that the configuration didn’t reach the desired point.

On the arm end effector, a multi-spectral camera will be mounted in order to conduct early detection of biotic and abiotic stresses in specialty crops, for this purpose, the image orientation is not important, therefore, a rotation joint on Z-axis was added to the end of each arm.

## Optimization

As said before there are 1,695,044 configurations. with an average simulation run time of 15 seconds per configuration, it will take about 300 days to simulate all the configurations. Therefore, an optimization method was developed in order to find the optimal configuration in a reasonable time. In this case, the set-based concept approach, which combined configurations with the same properties in one design concept. In this specific problem, there are almost 800 concepts and it will be time-consuming to make a deep search of each concept. therefore, multi-objective evolutionary search for satisficing concepts based on dynamic window-of-interest developed. the aim is to reveal which of the concepts have at least one solution with a performance vector within a dynamically changed window-of-interest. In the first part, a predefined computing time allocated to find satisficing concepts that would be explored more deeply in the second part of the optimization.

In the second part, the concepts that satisficed in the previous part are going to be explored more widely and to find the Pareto front of the problem by using a multi-objective evolutionary algorithm.

### Objectives

The Objectives of the optimization problem selected from the indices from the previous section and are the following:

* f1 🡪 Max manipulability [0-1]
* f2 🡪 Min Z (Mid-Range Proximity) [0 -0.5]
* f3 🡪 Min Degrees of Freedom [4 - 6]

This optimization problem is mixed with minimum and maximum objectives, to change the problem to be minimum problem f1 objective has been changed as follow:

f1 🡪 Min (1 – manipulability) [0-1]

### Independent variables:

The independent variables of the optimization problem are:

* X1: Joints Types: array [Roll, Pitch, Prismatic]
* X2: Previous ax: array [X, Y, Z]
* X3: Links Lengths: array [0.1 ,0.4, 0.7] (meters)
* X4: Number Degrees of Freedom: Int [3, 4, 5, 6]

### Constrains

The constraints of the optimization problem are derived from the assumptions from the simulator section and are the following

1. First joint is rotational along Z axe: X1[0] = Roll, X2[0] = Z
2. First link length = 0.1m : X3[0]=0.1
3. Total length of all the links > 1m : Sum (X3) > 1
4. 2 adjacent prismatic joints must be perpendiculars: If X1[i-1]==Prismatic X1[i]==Prismatic than X2[i]!=Z
5. No more than 3 prismatic joints: \*\*\*\*\*\*
6. After Roll joint can’t be Roll\Pitch joint in the Z axe: If X1[i]==Roll and (X1[i+1]==Roll or X1[i+1]==Pitch) than X2[i+1]!=Z
7. After Roll Joint or the following joints sequence [ Roll -> Pris Z] the next joint wont be in the X axe: if (X1[i]==Roll or (X1[i-1]==Roll and X1[i]== Pris and X2[i] == Z)) than X2[i+1]!=X
8. Joints limits: for Roll\ Pitch [0-360°] and for Pris [0 – 2\*link length]
9. Number of detection points: 4
10. Success: need to reach to one of the 2 top points and to the middle and the lower points

### Concepts

Each concept is defined from the following variables:

1. DOF – Number degrees of freedom of the configuration
2. Pitch joints – Number of pitch joint in the configuration
3. Long Link - Number of long links (0.7m) in the configuration
4. Acc length – Accumulated length of all the links of the configuration. This variable is defined as a range of distances
5. # of Parallel axes Y – Number of parallel axes along world Y ax in a row
6. Longest Link – What is the longest link in the configuration
7. P/R ratio – The ratio between Prismatic joints to Revolute joints (Roll\Pitch)

All the variables and their possible values are presented in Table 1.

Not all the combinations between the variables are possible. For example, it's not possible to concept to be with 4 DOF and 5 Pitch joint.

The total amount of concepts that received is 794 when there are concepts with 1 configuration and concept with up to 68520 configurations.

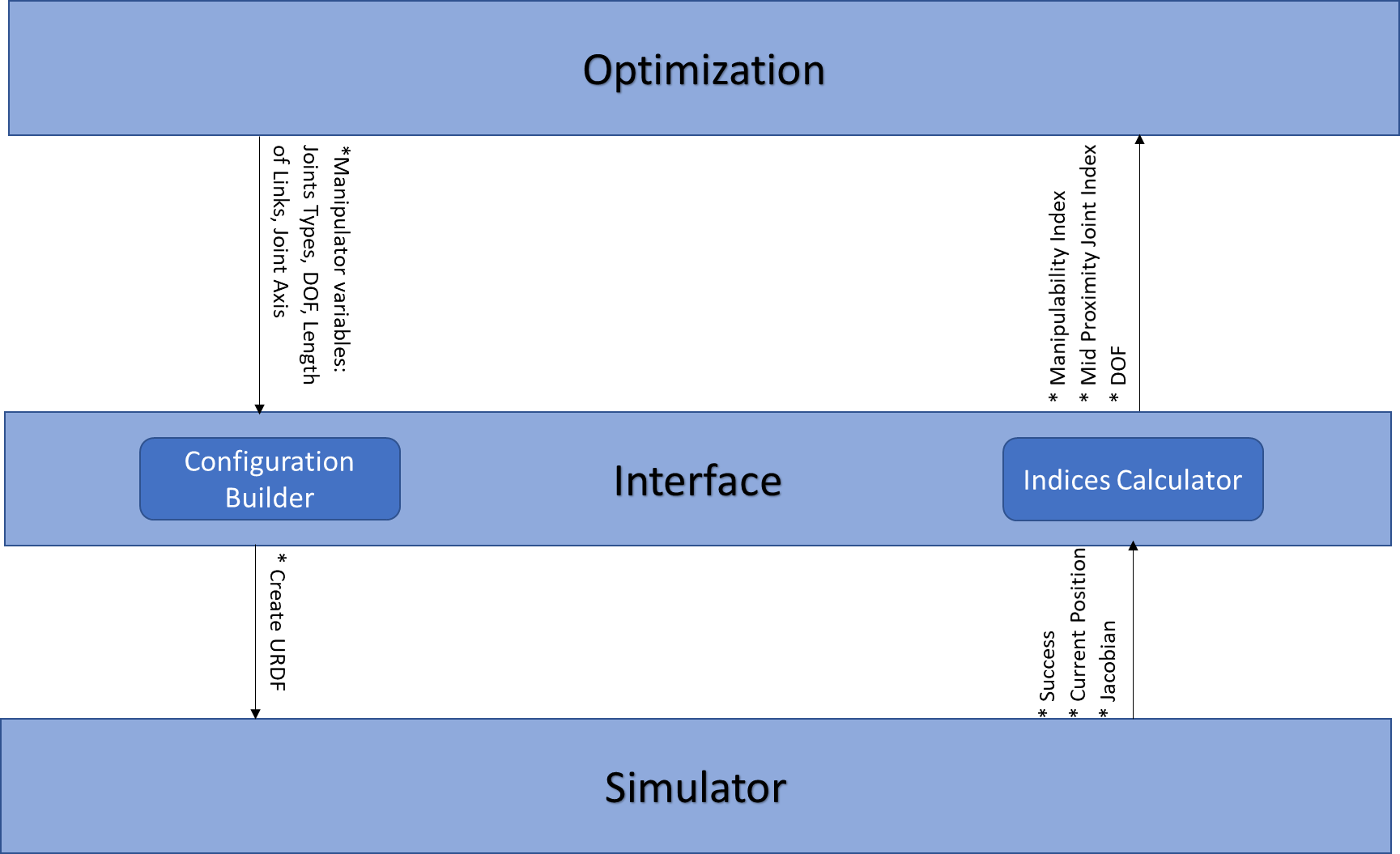
Table 1 - Concepts



## An interface between the simulator and optimization algorithms

In order for the simulator will be able to communicate with an optimization algorithm, an interface was built in Python. The interface gets from the optimization algorithm the variables selected by the algorithm to be simulated (DOF, Joints Types, Joints axes, Links Lengths) and creates from this data a URDF file which contains the kinematic and dynamic representation of the manipulator. After creating the URDF file the interface enters it with the predefined detection points into the simulator and gets in return if the configuration succeeded to reach the desired points and if it’s succeeded what was the Jacobian and joints position at every point.

The interface calculates the manipulability index and the mid proximity joint index and returns it to the optimization method to find the optimal manipulator. In Fig \*\*\* it can be seen a scheme of the interaction between the interface and the simulator and the optimization method.



## Build a preliminary Window of Intrest

As described before, the optimization is done in 2 steps: the first step is to find several concepts from the 794 concepts that satisfying. The concept that satisfying is a concept that is at the Window Of Intrest (WOI). In order to start this step in reasonable WOI, random configurations were simulated. The configurations were selected from all the concepts in fair selection. The selection is done as follows:

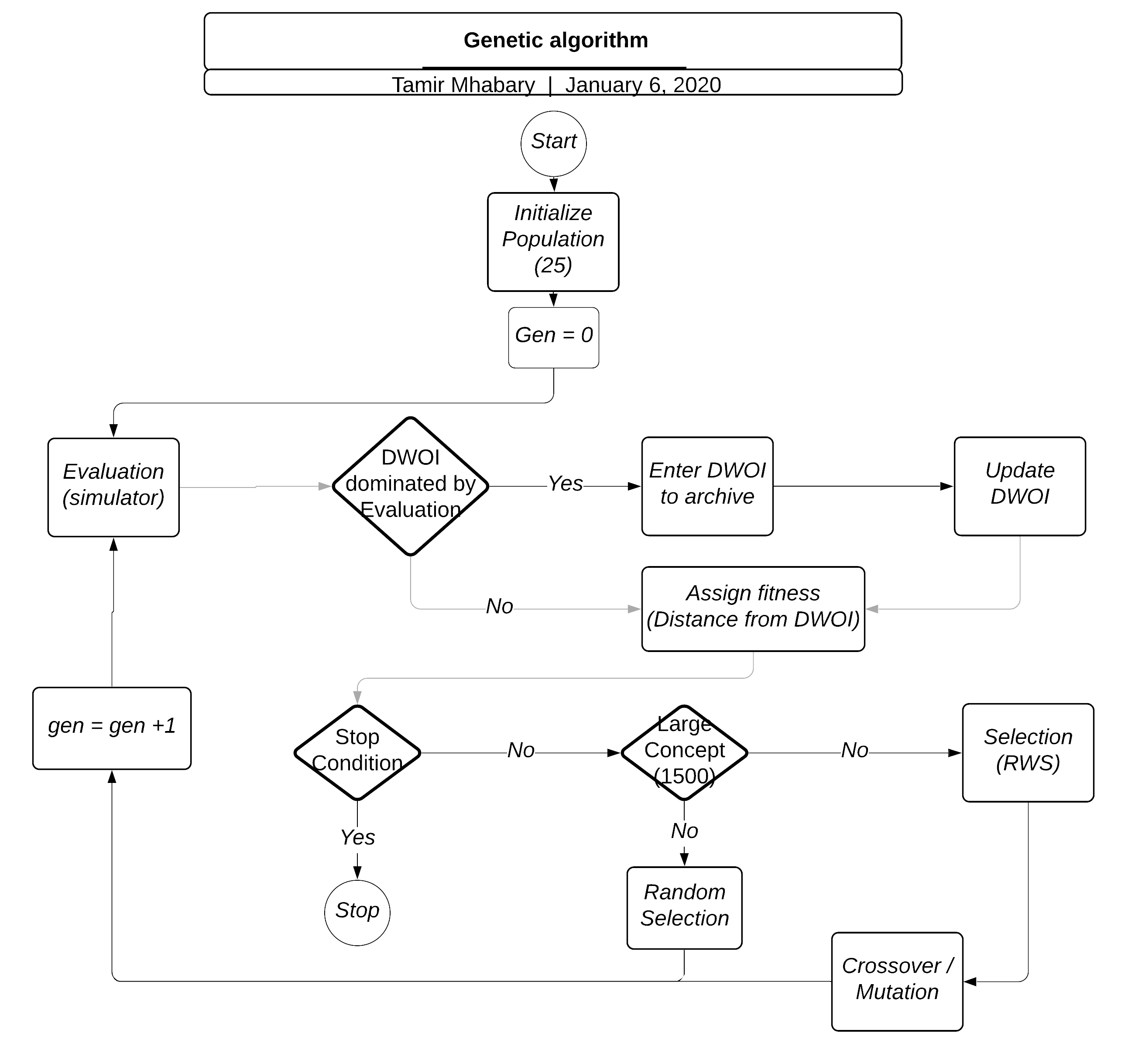
* One week of computer time (there were 1.5 available computers) – each configuration taking around 15 seconds to simulate so 60000 configurations selected
* From concepts with less than 25 configurations, all the configurations selected
* From concepts with more than 25 configurations, selected 25 configurations or 3% of the configuration, what higher
* Select all the non dominated solutions to create WOI (can be seen in fig \*\*\*)
* The WOI can be built from several concepts, mustn't be from one concept

## Evolutionary optimization with Dynamic-Window of interest

In this type of algorithms, the WOI is dynamic (DWOI), meanings that WOI updated during the processes and continues to approach the origin. The way of calculation the initial WOI described in the previous section. The crossovers and mutations are done only inside each concept, aren’t done between concepts. The grading inside each concept is done by calculating the distance between each configuration to DWOI. The configuration(s?) which are closest to the DWOI continue to the next crossover. If one or more of the configurations is non-dominated by the DWOI they will enter to the DWOI and all other configuration in the DWOI which dominated by the new configurations will be excluded from the DWOI and will enter to the archive. Concepts with a small number of configurations the selection will be randomly and concepts with a large number of configurations the selection will be done by the genetic algorithm.

To add: \* Memory allocation \* Stopping criteria \*Algorithm tree

To do? surrogate model



## Results analysis

\* Analysis the front

\* select concepts to continue with

\* Select indices for the last part: HV, IGD, etc.

## Find Pareto front using a multi-objective evolutionary algorithm

\* Select algorithm

\* Compute Indices

\* Show results