Calibration and validation of an agent-based airport security checkpoint model with heterogeneous agents

Thesis

R.P.M. van der Sommen

TU Delft

Calibration and validation of an agent-based airport security checkpoint model with heterogeneous agents

Thesis

by

R.P.M. van der Sommen

in partial fulfillment of the requirements for the degree of

Master of Science
in Aerospace Engineering

at the Delft University of Technology,

Student number:	4187040	
Project duration:	March 21, 2018 – July 5, 2019	
Thesis committee:	Prof. dr. ir. R. Curran,	TU Delft, chair
	Dr. O.A. Sharkanskykh,	TU Delft, supervisor
	MSc. S.A.M. Janssen,	TU Delft, daily supervisor
	ir. M. Schuurman,	TU Delft, external board member
	Bsc. A. Dilweg,	Rotterdam The Hague Airport, external board member

TU Delft

Contents

List Of Acronyms

Abm	Agent-based modelling
AATOM	Agent-based Airport Terminal Operations Model
ETD	Explosive Trace Detection
IATA	International Air Transport Association
MiVa	Handicapped ('Minder Valide')
JATM	Journal of Aerospace Technology and Management
RTHA	Rotterdam The Hague Airport
SC	Security Checkpoint
SL	Security Lane
WTMD	Walk-Through Metal Detector

1 Introduction

With an expected doubling of the number of air travel passengers in the coming 20 years, an increase in capacity by airports is required [6], [14], [15]. Constraints can be imposed by the lack of existing space in airports or the inability to increase this space to grow, as well as goals to minimise costs. In order to be able to meet these challenges, airport functioning requires analysis to get insight where processes can be improved [4]. As real-life experimentation of different concepts to change the current operation can be costly or infeasible al together, simulation is an indispensable tool. A relatively novel approach for analysing complex sociotechnical systems such as airports is through agent-based models (ABMS) and modelling. As opposed to existing methods based on operations research [35], in Agent-based modelling, the physical space and individual agents (passengers with their properties, behaviour, and interactions) are simulated [31]. The individual agent behaviour combined eventually leads to emergent - macroscopic - behaviour. These movements, actions, and interactions of each passenger can be observed throughout the simulation while being traced [28]. This is a more true-to-nature representation of the system compared to the classical paradigms, as agent flow, congestion, and agent interaction can be studied (graphically), which might lead to improved security checkpoints designs and/or procedures. Examples are conveyor belt length optimisation, improved routing, or updated passenger check handling. In order to be able to asses these alterations based on simulations, it is vital to have confidence the model gives meaningful results; are the results valid? Answering this question is formalised in this thesis' research question:

What is the validity of the predicted performance of a heterogeneous populated agent-based model inside the initially intended range compared to experimental data?.

In each airport, the behaviour is influenced by certain processes passenger need to go through before one is allowed on the flight. This includes, the check-in, security checkpoint, and the actual boarding itself [4], [26]. It is therefore of importance that each of these essential processes are modelled accurately as to be able to get simulations results which represent reality [42]. Of particular interest is the security checkpoint, since this is a complex, time consuming, and costly operation, while also being an operational bottleneck [4]. Real-life assessment of operation performance - average passenger security checkpoint time and security checkpoint throughput - is time consuming and costly. Testing of changes in a bid to improve performance can be impossible all together. Computational simulations can provide a solution to estimating expected performance differences due to changing operations.

To this end, an existing agent-based simulator is used for airport terminal processes: Agent-based Airport Terminal Operations Model, AATOM [26]. Apart from describing a security checkpoint, it also allows for the behavioural analysis of a heterogeneous population in the context of airport terminal operations. Not all passengers behave similarly when going through a security checkpoint. Age, goal of travel and group composition can influence the behaviour - and performance when moving through the security checkpoint - significantly [44].

2

Literature Review

In this Chapter, existing literature and insights are presented which serve as an essential foundation in trying to validate AATOM. This is essential to ensure meaningful result are obtained. This is a a concise version from a separate literature study performed in the beginning of this master thesis study, as the model first needs to be able to describe existing operations before changes can be considered for improving the security checkpoint performance. Topics chosen to be included in this literature review are the workings of a security checkpoint 2.1, the use of an agent-based model 2.2, the chosen simulator AATOM in particular and its characteristics 2.2.3, and finally a framework describing the efforts required to get the model to represent reality to a sufficient degree.

2.1. The Security Checkpoint

Firstly, the airport terminal processes underlying the model are discussed. One has to take into account all the relevant processes. Furthermore, it is tried to be as simple as possible [36]. This results in the inclusion of the most important processes into the simulation tool while omitting the non-relevant processes, based on expert knowledge. The scope of possible processes to be included are the ones encountered when a passenger first enters the queue leading to the security checkpoint - Section 2.1.1 - and stops when all the luggage has been reclaimed in Section 2.1.6. A graphical representation can be seen in Figure 2.1, where the passenger enter the security lane from a queue to the left while it moves to the gates to the right.

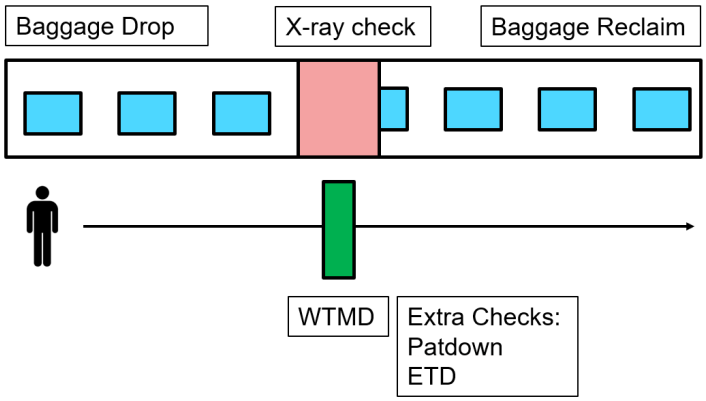


Figure 2.1: Graphical representation of a security lane, which form a security checkpoint

2.1.1. Queue

The queuing is the first process encountered by the passenger. Observe that this is not a process a passenger or security officer can influence directly. This is a result of the arrival rate of the passengers, as well as the throughput of the entire security checkpoint system that will result in queue forming, dissolving or maintaining a 'status quo'. Hence, it is not included in the Figure 2.1

2.1.2. Baggage Drop
When first in line, a passenger can enter one of the security lanes. When a lane has been chosen, the passenger is required to put all the relevant baggage and belongings in boxes. This includes all carry-on baggage, as well as clothing, shoes, belts, and other items as defined by IATA regulations [23]. This baggage drop is in preparation of two processes that will be discussed next, being the *x-ray scanner* and the *walk-through metal detector*.

2.1.3. Walk-Through Metal Detector
A magnetic field is used to determine if a passenger carries metallic objects through the metal detector. Some detectors can indicate in the case of a positive test result the area/zone on the person in which metal has been observed. This zone-indication helps the security officer in limiting the area required to check on the passenger. When an alarm is raised by the WTMD, a standard additional check in the form of a pat-down is performed. A major drawback of the WTMD is that it is only able to detect metals [49].

2.1.4. X-ray Scanner
While the passenger is checked using the walk-through metal detector, the relevant baggage and belongings dropped during the *baggage drop* is analysed using a x-ray scanner. The output is a coloured 2D image outlining individual objects in the examined x-ray box, where a security officer determines if the objects do or do not comply with regulations specified for carry-on baggages. The colouring of the objects is such as to assist in assessing both the shape and the material of the objects under investigation. If a (possible) malicious object is detected, the baggage is investigated further before the passenger is allowed to reclaim their belongings and leave the security checkpoint.

In a bid to improve effectiveness of the x-ray, a Threat Image Projection (TIP) program was implemented. An image of a threat is superimposed on the actual image of the scanned objects. It is a way of measuring the performance of the security employee, as it can be registered how many intentionally generated TIP positives have been recognised. Furthermore, it helps in keeping employees attentive [4].

Some difficulties associated with 2D x-ray images is that the orientation and superposition of objects complicates the correct identification and classification of objects analysed. Furthermore, bag complexity may clutter and obfuscate the images.

2.1.5. Standard Additional Screening
If the walk-through metal detector keeps detecting the presence of metal objects on the passenger - after the visible causes for a positive test have been removed - an additional screening of the passenger is conducted in the form of a pat-down. A security employee follows the contours of the passenger in a predetermined manner to exclude objects are carried on the body. Objects found are inspected. If none are found and no further suspicion remains, the passenger is let through.

Garments associated with religion or otherwise culturally dictated might be required to be subjected to a manual search. This is done since loose fitting would allow for easy concealing of objects not allowed onboard.

If the x-ray scanners detect the presence of objects that do not comply with regulations, a security employee searches the belongings until the object is found which matches the description as seen on the x-ray scanner. A visual inspection or a second x-ray scan is used to determine if the object might be brought along on board or not. If not, the object is thrown away.

Finally - at random - the walk-through metal detector selects passenger for an additional Explosive Trace Detection (ETD) procedure. The selected passenger and its belongings are both sampled using a rod which is inserted into an ETD-scanner. The sample is checked for chemical substances associated with explosives or narcotics. These scanners are extremely accurate, when the sample is of sufficient quality (enough material is found for analysis) [49].

Downsides associated with the ETD-check is that it is labour and time intensive to perform the test. Furthermore, only a portion of the passenger are subjected to the test, ranging form 10% in the European Union

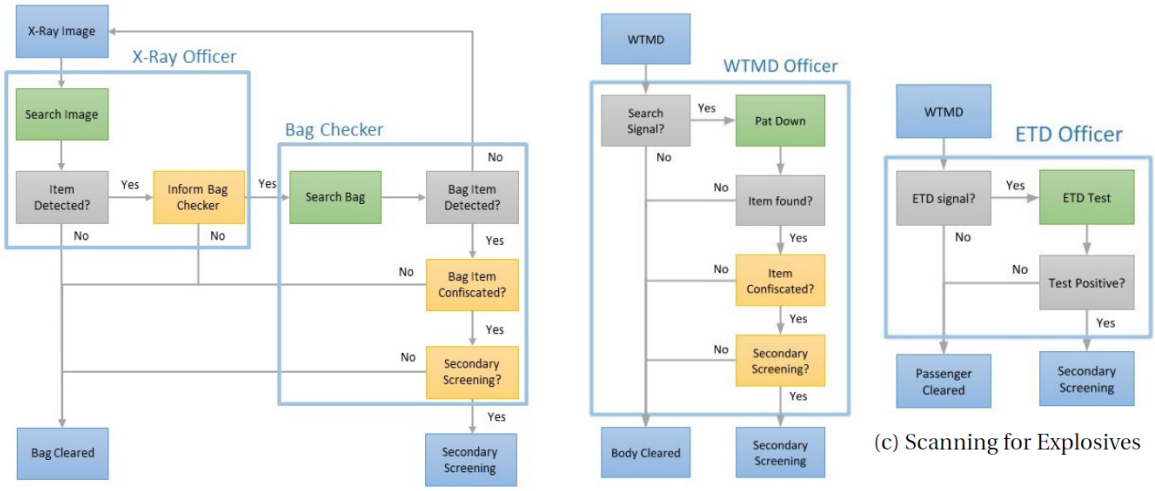
to 60% in the United States of America. Also, false positives are an area of concern, as many household materials can be either used for everyday use or, when mixed to some specifics, can be transformed into an explosive substance [49].

2.1.6. Extraordinary Additional Screening

After performing all the checks mentioned above, some safety concerns might still exist about the passenger under investigation. A more in-depth investigation can be done by taking the passenger to a separate office near the security checkpoint. No provision is currently present to accommodate this eventuality in AATOM [49].

2.1.7. Flow Diagram Screening Process Security Checkpoint

The aforementioned processes can be graphically linked using a flow-diagram, as proposed in [49]. The following flow-diagram is applicable to a regional airport, such as used for the calibration and validation of the AATOM model.



(a) Scanning Carry-On Luggage

(b) Scanning Passengers

(c) Scanning for Explosives

Figure 2.2: Graphical representation security checkpoint screening process of a regional airport [49]

2.2. Agent-based Modelling

With the workings of the security lane and checkpoint known, the translation to an agent-based model can be investigated. As mentioned before, an existing model - AATOM - is used in this study which will be subjected to calibration and validation efforts in a bid to *'substantiation that a computerised model within its domain of applicability possesses a satisfactory range of accuracy consistent with the intended application model'* [42], [43]. For this thesis, the movement and processing of passengers through a security checkpoint is assessed. But first, it should be stressed as to why agent-Based Modelling is apt for the case under investigation.

2.2.1. Agent-based Modelling Usage

A security checkpoint is a complex, multi-actor, intelligent, distributed systems. Agent-based modelling can be used to represent such a system. In the model, an environment is defined; for this study the security checkpoint at an airport. In the environment, agents are active. An agent could be defined as an autonomous, computational entity that perceives its environment through its sensors and acts upon its environment through effectors. An agent can therefore represent both human and non-human entities. The interaction of the agents - and with each other - in the physical environment form the basis of the simulation. How the interaction between agents occurs is the result of the behaviour given to individual agents. This can range from relatively simple rules to complex models trying to address cognition. It is important to observe that these rules are specified for autonomous acting on part of the agents [31]. It is this property of decentralised agent behaviour resulting in emerging overall properties that makes agent-based modelling apt for the simulation

of a security checkpoint.

This behaviour and interaction between agent types give the model and the resulting simulations its observable behaviours, patterns, and structures. These result not through strict programming, but rather emerge, as local behaviour and interaction at the individual level aggregate to global structures and patterns. Since the rules followed during interaction can be different for each agent within the system, a heterogeneous population can be created. Different concentrations of agents with different rules in the simulation therefore can lead to different interactions and emergent behaviour, patterns, and structures. It should be observed that - since each agent has a unique identifier and is therefore uniquely traceable during the simulation - individual behaviour or subsets of agents can be studied as well [31].

With the versatility of agent-based modelling stressed, a closer look needs to be taken at how this modelling paradigm is used to represent the security checkpoint. It is essential to demarcate the scope as a near infinite level of detail can be attained if unlimited resources are available. Since resources are always limited, the scope of the model has to be limited as well, while remaining detailed enough to be of practical use. The evaluation of the level will be done using AATOM (the existing agent-based model used in this study).

2.2.2. Agent-based modelling Level Of Detail

The set of rules can vary in complexity and consequently can describe processes with different levels of detail. Three levels of detail are considered, termed 'aggregation level', respectively level I, II, or III. Each level is a more in-depth improvement with respect to the previous level, as can be seen in Figure 2.3.

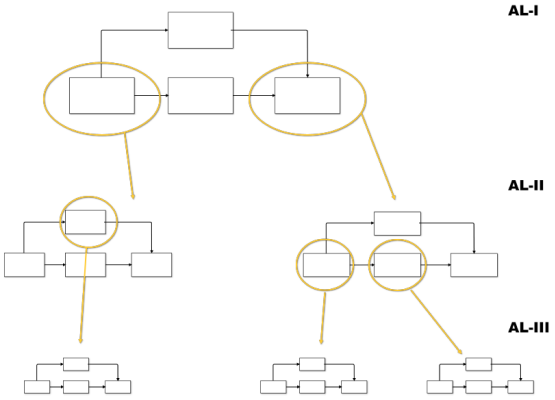


Figure 2.3: Graphical representation relation between 'aggregation levels' AL-I, AL-II, and AL-III [38]

The first and most top level of the three levels is the 'black box'. In this the entire security checkpoint - checkpoint in Figure 2.4 - is seen as only one input and one output. When a passenger enters the black bo, it exits the black box after a certain amount of time without the causes for this delay being specified. The second aggregation level does acknowledge the individual process within the level one 'black box'. Now the behaviour of the passenger agents on the different processes can be observed. This leaves the third and last level. Herein, the (cognitive) processes resulting in the observed behaviour - level two aggregation - is attempted to be modelled [38]. The distinction between the different levels is made since a time consuming level III simulation might not be needed, and simpler level I or II suffices. Time consuming refers to both creating the model with this level of detail and assessing the results, as well as the additional runtime for running the model. Furthermore, during model development, complexity can be added as the model matures. Starting from a level I representation, paving the road for level II and subsequently level III implementation.

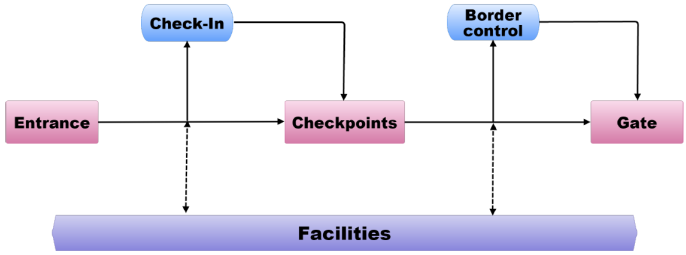


Figure 2.4: Graphical representation 'aggregation level 1' of a regional airport such as Rotterdam-The Hague Airport [38]

Since the level II aggregation level is a more detailed representation of the processes defined in level I - as an example - a closer look is taken at the security checkpoint. This can be seen in figure 2.5. Here, processes discussed in 2.1 can be seen to reappear as level II aggregation does consider individual processes within the 'black box' security checkpoint.

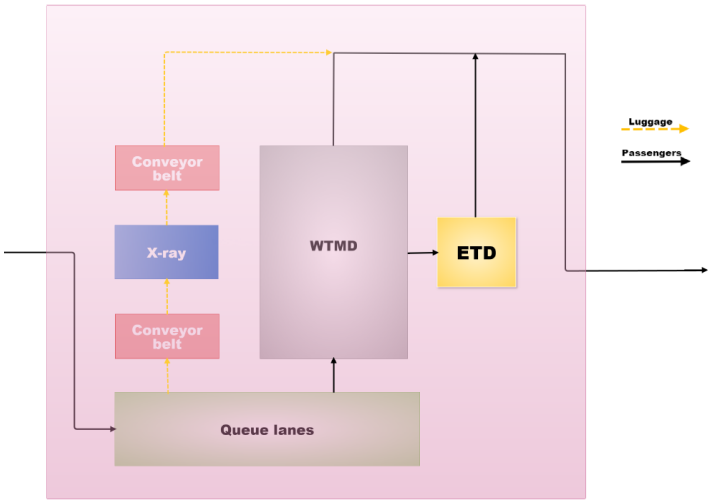


Figure 2.5: Graphical representation 'aggregation level 2' of the security checkpoint [38]

Each of the three aforementioned aggregation levels - and hence level of detail - consists of three parts. These parts are the static structure, relevant processes, and organisational layer. The static structure defines the physical constraints, such as walls, doors, and floors. It is therefore the airport with the interior design present. The relevant processes are the (mathematical) representations and interactions between the different agents. The third part is the organisational layer. In this layer, the policies are defined. The policies can be seen as a 'rule book'. If a certain condition is registered, predetermined action is taken. For instance, when a certain queue length is reached, a new security lane must be opened [38]. With the processes to be included and their level of detail in AATOM known, a closer look has to be taken how this is implemented in the model. This will be the topic of the next Section.

2.2.3. Agent-based Modelling AATOM

The theory and ideas are implemented in a model. AATOM is able to represent the entire passenger movement through an airport, ranging from arrival to departure [26]. As mentioned before, the scope of this master thesis study is calibrating and validating the security checkpoint model. Hence in this subsection a closer look is taken only at the specification of the AATOM parts relevant for the security checkpoint. A formal description of the relevant parts of the model can be found in Appendix A. As this is rather lengthy, a more concise, informal description is given next based on the formal description. As described in the AATOM documentation [26], the security checkpoint is one of ten areas accessible to agents. In these area's, physical objects are placed to form the environment. This includes the queue separators used to implement the snaking of the queues, as well as the layout and devices used in the security lanes. These sensors forming the lanes consists of x-ray machines and WTMDs. Also, places for operator-agents are reserved. These parts together form the environment of the security checkpoint in the model.

Three agent types operate in the environment. Passenger agents represent the passengers at an airport. Furthermore, operator agents are modelled. These act as employees working at the airport. At the security checkpoint, these operators represent among other things employees tasked with the boarding pass check, operating the x-ray, and performing the ETD-check. Thus, both passenger and employee agents are humanoid representations in the form of agents and are henceforth described as human agents. Finally, orchestration agents can be modelled. Either in the form of a human or non-human entity, are capable of steering operator and/or passenger agents and adjust the environment to achieve specific goals. The relation between the three agent types can be seen in Figure 2.6 [26]. The human agents are composed of a three-layer structure,

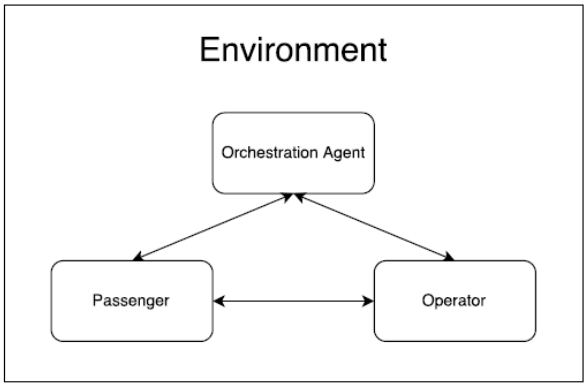


Figure 2.6: Interaction between passenger, operator, and orchestration agents [26]

depicted in figure 2.7 and can be viewed as the 'human brain', resulting in the agent's behaviour (which is tried to resemble human behaviour). The *strategic layer* determines the goals and updates of the belief module. Furthermore, reasoning is done in this layer and includes performing analysis resulting in planning and decision-making. The middle layer - the *tactical layer* - is tasked with actuation of activities. This includes interpretation of observations and route navigation. The third and final layer - the *operational layer* - is responsible for interaction with the environment and interaction with other agents. Input is sensory data from the agents sensors. Output are the actions, defined by the input, going through the tactical, strategic, and tactical layer, for eventual execution by the operational layer. For a more in-depth analysis of the layers, the interested reader is referred to [26].

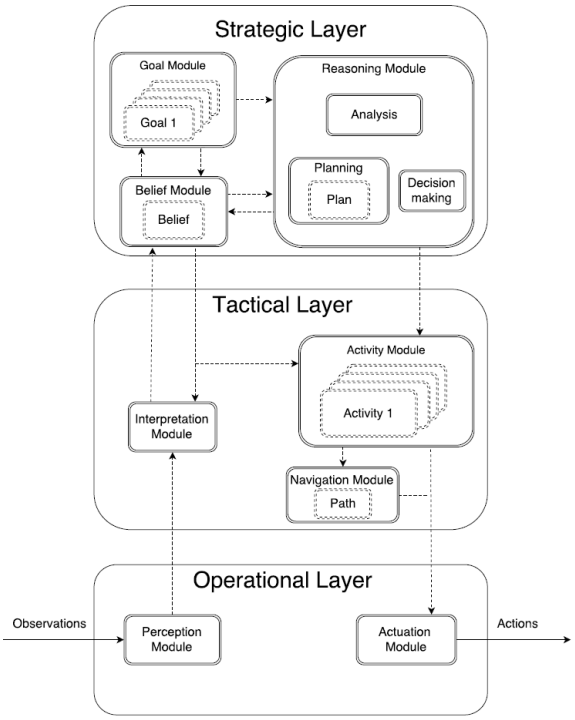


Figure 2.7: Schematic layering of modular structure of human agents. Solid lines represent input-output relations. Dashed lined indicate interaction between modules. Dashed boxes represent memory within a module [26]

Before concluding the AATOM description, it is essential to get a better understanding in how the implementation of the security checkpoint in level II aggregation is done. This will expose the options, tools, and parameters available for manipulation and calibration. It can be read in [26] that the processes within the security lane - both the sensors, as well as interactions with operator agents - is modelled through a waiting time where the agent stands idle before continuing through the security checkpoint. In AATOM, this idle time can be specified for the baggage drop and baggage reclaim, see Figure 2.1. This waiting time is a direct result of the action being undertaken. For instance a waiting time resembles the time required to perform the luggage drop. In practice, time is required to take of ones jacket, clear ones pockets an putting it into x-ray boxes. These actions are not modelled directly, but represented as a waiting time in which the agents stands idle, as in practice, the agent cannot continue in the security checkpoint until these tasks have been completed. This luggage drop time can be supplemented by an interaction with a luggage drop security operator, which represents the additional assistance given to assure the luggage is dropped of correctly. Waiting times are used as only the observed behaviour is tried to be represented of aggregation level II. Put differently, the cognitive processes leading up to behaviour observed at the processes within the security lane are not modelled. Merely the consequences thereof - the time it takes to go through a process - are registered and used in the model. This is in line with an aggregation level II model [38]. These waiting times are implemented in AATOM as being a random variables with distributions. Hence, a passenger is given a waiting time for a process that is allowed for by the specified process distribution. This distribution - by definition - also specifies the probability (and hence the frequency of this value occurring) of the waiting times to agents. These distributions are found through the analysis of empirical data.

It is these waiting times specified by distributions that play a pivotal role in being able to tune the model and try to make the simulations behave similarly to reality. This tuning process of constructing these distributions and the analysis of how well the tuning is done is the subject of the next Section.

2.3. Calibration & Validation

When a model has been constructed, it is vital to asses if the intended goals and requirements taken as a starting point are met [28], and it can fulfil its intended use. This is done through calibration and validation. Validation can be defined as 'substantiation that a computerised model within its domain of applicability possesses a satisfactory range of accuracy consistent with the intended application of the model' [42], [43].

Based on data gathered, the model input parameters must be manipulated through distributions such that model validity can be achieved. This process is called calibration. Parameter setting is done through data-analysis. Iterative loops through assessing the output given a certain input parameter setting might result in different analysis of available data. This in turn might alter the input parameters to get the model output closer to the observed real-life output.

While validating - as well as calibrating - it is important to keep using 'common sense'. When deploying and implementing intricate techniques, the results might seem useful from a theoretical point of view, while common sense would completely discard the findings. This is formally called *face validity* to ensure 'logical' results are found. When given to an expert, does he or she find the results reasonable [28]? Although it might seem obvious and therefore superfluous to mention, qualitative decisions are inevitable where this common sense is vital. In the next sections, a closer look is taken at the philosophies, methodologies and techniques for a structured approach to the calibration and validation efforts.

2.3.1. Calibration & Validation Framework

With the importance of calibration and validation efforts stressed, a plethora of frameworks to structure these efforts have been formulated. Examples are *indirect calibration*, *Werker-Brenner*, *History Friendly* [13], VOMAS [39], *model comparison* [51] and many alternatives. Although all tailored to specific use cases (primarily not for the agent-based modelling paradigm), many similarities can be distinguished.

In general, five steps (1,2,4,5,6, not 3) can be identified which are present in many - if not all - frameworks, being:

1. Face validity
2. Internal validity
3. *Tracing (Agent-based modelling specific)*
4. Calibration (using experimental data)
5. Sensitivity analysis
6. Statistical testing on output and experimental data

It should be observed that step 3 is italicized, as this is a unique addition for agent-based modelling. Agent-based models often have an ability to graphically show the movement of the agent through the simulated environment. During agent tracing, the movements are followed and assessed if the behaviour is in accordance with the programmed implementation; internal validity. Furthermore, it can serve as an aid in face validation, as an expert can say something about the observed movement of the agents, and if this is in accordance with reality or its expert knowledge [51]. Xiang [51] also concludes that the large similarities between steps to be completed also allow for the use of techniques used within in the classical (deterministic) domain. Finally, since an existing model is used, steps one and two have been done already. Hence, the remaining steps three through six are of interest during this master thesis study.

2.3.2. Calibration & Validation A Data Driven Approach

The aforementioned steps in the framework should be seen in light of a data driven approach. Here, data is the starting point. This is a different approach compared to more traditional model development. One such example is an analytical queuing model [9], [35], [36]. In an analytical model, an exact result can be obtained. Most models however, are too complex in order to be able to be solve analytically, and simulation is required [30]. Also, accounting models can be used. Similar to analytical models - being deterministic - rules rather than mathematical expression are used to describe the state of the system. Finally, time-dependent systems

are considered. Using dynamic mathematical equations, an event-based model can be constructed. This also allows usage of stochastic processes in the model. Steady-state results can be found using Monte Carlo methods [36], [50]. The aforementioned model types have all one thing in common. The equations describing the model are pre-defined and tuned using data. With the development of increasingly powerful computers, another approach has been under development, being the data driven approach.

A data driven approach works the other way around. Now, data dictates the how and which equations are formulated, based on the available data. The area of computational intelligence - such as machine learning - is used employed to formulate relations. Without regard for physical processes - a representation is sought to describe the link between input and output. Examples of statistical techniques used are rating curves or regression models. Machine learning techniques examples are artificial intelligence, data mining, and knowledge discovery in databases [46].

Irrespective of how the relations are constructed or data is used, it should be tested if the resulting models give correct results. And if errors are found, if these can be reduced or eliminated. This is done in calibration and validation.

2.3.3. Calibration & Validation Techniques

Calibration is the fourth step in the Calibration & Validation Framework 2.3.1. Klügl [28] defines the process of calibration as 'parameters have to be set in a way that a structurally correct model produces a valid outcome' [28]. Using iterative loops of calibration and model parameter tweaking, validation is done, which is the final step in the Framework 2.3.1. Three types of validation are distinguished [11]. *Descriptive validation* is associated with establishing whether the model is able to explain phenomena or organise information in a meaning full way. One can think of this as follows: C happened because A and B interacted unexpectedly at time t on location x. Since the model used does not have such properties, this type of validation is not performed.

The second validation type is *structural validity*. This validation is executed to determine if the relevant processes, policies and environment with the correct associated attributes are present in the model. It can vary for different levels of detail [28].

Lastly the *predictive validity* can be assessed. Put simply, is the model able to estimate the future - based on the input data and the intended goal of the model - with an acceptable level of accuracy? It is this predictive validity, combined with the structural validity, that will be investigated further [28].

Thus not only calibration, but also validation efforts rely largely on the availability and analysis of data. For validation, the question of interest to be answered is 'substantiation that a computerised model within its domain of applicability possesses a satisfactory range of accuracy consistent with the intended application model' [43], [42].

As stated, experimental data is the foundation for both constructing the input distributions, as well as assessing the validity and level of being able to represent reality based on the model outputs. Hence, firstly, the (empirical) data should be examined.

Data At First Glance

Finding the *central tendency* of the recorded random variables of a certain phenomenon is a starting point in getting an idea how much time a process takes. The *mean* is the average value of the entire set. The mean is sensitive to outliers (especially when large in magnitude). Therefore, the *median* should be considered also. This is the smallest value of an experiment x for which the cumulative probability of this value is for the first time larger or equal than 0.5, $F_{X_i}(x_{0.5}) = 0.5$. Also, the mode should be discussed. The *mode* is the value of the experiment x which is the most likely to occur, maximise $f_{x_i}(x)[p_{x_i}(x)]$.

It can be useful to asses the central tendency graphically. To this end, the data can be plotted in a *histogram*. Herein, the data are grouped into bins - group the available data points into a fixed set of discretisations - and count the number of data points that fall in each bin. The resulting frequency distribution - which is a synonym for histogram - lets one visually see if certain values occur significantly more than other, as well as

if the values are packed tightly around this value with thin tails (*leptokurtic*) or packed loosely with fat tails (*platykurtic*). Furthermore, it can be seen if the values are distributed evenly around the central tendency (*normally distributed*), or grouped asymmetrically - *skewed* - towards the lower end (*positive skew*) or the higher end (*negative skew*) [21]. When the data is assessed graphically, a more formal way of describing the distributions is required.

Data Distribution Fitting

The discrete - empirical data - needs to be transformed to a form which is accepted by the simulator. To this end, the discrete data points need to be converted to a continuous distribution. But what distribution type should be chosen? And what parameters are used? To this end, more formal mathematical techniques are employed and taken from the toolboxes and framework provided by Matlab. The parameters of the distribution are found using the 'maximum likelihood estimation' method. Next, each fit is assessed for how well the distribution represents the empirical data using 'goodness-of-fit' tests. The non-parametric 'Smirnov-Kolmogorov' test is typically used [21]. Here, the largest absolute difference between the experimental cumulative distribution function and the empirical cumulative distribution function is a measure how well the distribution represents the experimental data. The Anderson-Darling test is another viable test, which performs better near the tails [8], [21], [30], [40]. These test can only be used if a sufficiently large sample size is available. This can be tested by assessing the critical value using $D_{critical} = \frac{1.36}{\sqrt{N}}$ [32]. When this value is larger than the largest difference between both cumulative distribution functions the fitting score is eligible for use and can be used in deciding which distribution to be used. Although the data might be represented correctly, it is also vital to asses if the correct data is used for fitting.

Data Merging And Relations

It is might be vital to asses if data found might be combined or not. This as sometimes a separate dataset is too small to get statistically significant results if the critical value is not satisfied. To this end, datasets are to be compared using Mann-Whitney or Kruskal-Wallis tests [40]. It tests if the medians of two datasets (or more for the Kruskal-Wallis test) are equal, given the assumption the groups are independent. This is vital to asses if certain sample configurations are allowed for and are eligible for distribution fitting after the data has been merged. Not only assessment if data-groups can be combined is of importance.

Furthermore, it is essential to asses if observations of different processes might be linked. This is required to uncover relations of influencing factors. This might result in data-fusion through combining information - waiting time data - which forms the basis for different - improved - distribution fitting. A technique used often is the Pearson correlation matrix [30]. It gives a linearised correlation coefficient between selected parameters

When the input distributions have been set based on a meticulously selected datasets, it is important to asses what the effects are for the simulation. To get reliable results, multiple replications - with a different seed and hence differing slightly per run - of the simulation given the settings need be performed.

Simulations Getting Reliable Results

A trade-off between resources required to perform the replications and the confidence that can be placed in the result given the number of replications. To this end, a confidence interval can be used. Given a certain confidence requirement - typically Fisher's 95% - a range of values can be established in when replicated 100 times, 95 of the values found will be inside the range. This range decreases when the number of replications increases. Formally known, this range is called the confidence interval. The interval for a certain output can be computed as follows. The sample mean is found by finding the mean of the means, through Equation 2.1.

$$\overline{X}(n) = \frac{\sum_{i=1}^{N} S_i}{N} \tag{2.1}$$

The sample mean is used to construct the sample variance. This is the distance per mean with respect to the sample mean. This value is squared as to always get a positive distance and cancellation of distances is no longer an issue. This can be seen in Equation 2.2. Observe the denominator now used $N-1$. Since an inference is made about a larger population, a degree of freedom is lost; Hence, the reduction by a value of 1.

$$S^2(n) = \frac{\sum_{i=1}^{N} [X_i - \overline{S}(N)]}{N-1} \tag{2.2}$$

Finally, the confidence interval can be constructed. This is structured around the sample mean, where the sample variance is used to define the dispersion around this average. This can be seen in Equation 2.3. Observe the $t_{0.95}$ corresponds to the 95% z-score of the inverse cumulative distribution Student t-test value. This is used, as the central limit theory forces the values found per simulation tend to go towards a normally distributed distribution.

$$Confidence\ Interval = \overline{X}(N) \pm t_{0.95} \frac{S^2(N)}{N} \tag{2.3}$$

The sample mean and confidence interval play a vital role in assessing the simulation performance, as the performance indications can be compared to the experimental values [21], [30].

From Calibration To Validation

The tools and techniques presented in the aforementioned (sub)sections are the elementary building blocks in finding input values, assessing the results, tweaking the input values and re-assessing the results. Using these tools in iterative loops, it one tries to get the model to represent the experimental data and be able to show confidence in the results found though simulations using AATOM representing reality to a sufficient degree. Balci lists 44 techniques in his review papers [1], [2]. A subset if the techniques are presented here.

Informal techniques

Informal techniques rely primarily on human expertise. No rigorous mathematical techniques are used. Rather, human reasoning and experience is used to assess the results. These include audits among others checking face validity.

Dynamic Techniques

Now, a closer look is taken at (intermediate) results. Using black-box testing, it can be tested if input parameters are correctly converted to outputs, as intended. If errors are found - for instance due to an incorrect value setting of an input - debugging is employed to find and correct the error. If applicable, use can be made of the graphical capabilities -visualisation - of the simulator to trace real-time how the model behaves and spot errors.

Formal Techniques

Finally, the predictive validity is to be assessed. Statistical techniques such as confidence intervals, non-parametric goodness-of-fit test such as the Kolmogorv-Smirnov and the Anderson-Darling tests, non-parametric test of means, and Pearson correlation matrices can be employed [3], [21], [29], [40]. Using these rigorous statistical techniques for testing real-life data and the model generated results, mathematically, the accuracy can be calculated. Interpretation of the results found are needed for deeming the formal proof of correctness accepted or rejected. Acceptance is required to deem the models valid [3].

3

Research Objective & Methodology

In the literature review, the workings of a security checkpoint are discussed 2.1, followed by how this is translated to an agent-based model in Section 2.2. It is concluded by which steps need to be taken to make the model behave as intended within an acceptable range of accuracy which will be discussed in Section 2.3. Whilst performing the literature review, gaps in existing literature have presented itself and will be discussed in the next Section 3.1.

3.1. Research Gaps

A number of gaps have been defined which will play a pivotal role in this master thesis. These are:

- **Being able to simulate a heterogeneous passenger population:**
  Agent-based modelling allows for simulating passenger agents with different characteristics and analyse the behaviour emerging from their interaction.
- **Validation of a heterogeneously populated agent-based model describing a security checkpoint:**
  It was found that only a small percentage of constructed agent-based models are being validated. Validation of an agent-based model therefore is useful. Validation of a heterogeneous - performance based - security checkpoint model is novel.

3.2. Research Objective & Questions

From the gaps identified, a research question can be formulated:

How to evaluate the validity of the predicted performance of a heterogeneously populated agent-based model with respect to experimental data?

This rather involved research question is broken down into a set of subquestions. These will help in ultimately being able to postulate an answer to the research question. These subquestions are:

1. How can a performance-based heterogeneous population be defined?
2. How can the heterogeneous population from (1) be translated to the agent-based airport security checkpoint model?
3. How can calibration techniques be used to make the agent-based model from (2) behave similar to the experimental data for normal operations?
4. How can validation techniques be used to asses the level of similarity between simulated and experimental results?

3.3. Methodology

Subquestions help answering the research question. A methodology helps answering the subquestions, which are needed for achieving the research objective. As has been foreshadowed in the previous Section 3.2, an agent-based model is used, as well as experimental data.

3.3.1. Experimental Data

Experimental data is the backbone of linking simulations to reality. As stipulated in Section 2.2, the model incorporates some of the deemed relevant processes which can be observed at an actual security checkpoint. Gathering data at an operational security checkpoint allows for model analysis to define both model input and assess the simulated results with respect to measured results. This data-acquisition will be done at Rotterdam The Hague Airport, where anonymous passenger data is gathered through security footage analysis. An in-depth description of the recorded data can be found in Section 4.1. Based on this data, we tried to define metrics for defining a heterogeneous population. Furthermore, it will serve as the backbone in shaping the (heterogeneous) population and translate this information to AATOM.

In the litarture review in Section 2.3.2 a data driven approach is described. It should be noted not the full spectrum of the data driven approach to modelling is used. The proposed approach is therefore a hybrid solution, where the model - AATOM - is constructed using a knowledge-based approach, where data-mining is to provide the mathematical representation of the processes, where interaction of the agents are responsible for the emergent behaviour.

What is meant is the use of areas found in data driven approaches and the observed data complements the knowledge-based model, AATOM. Although the relations are defined, the representation of these relations is to be defined through data-analysis of empirical data. This extraction of mathematical representations of the identified relations is done through data-mining, where it is tried to uncover valuable information from the data-set which might be concealed at first [16].

3.3.2. Data Analysis

The data will be subjected to both qualitative and quantitate analysis. Not only statistical analysis techniques are employed. Also, informed decisions and assumptions need to be made in addition to the statistical techniques in order to be able to manipulate data to be of use and in selecting statistical techniques. This will be the topic in Section 4.2. The translation of the results found to the agent-based model is discussed in Section 4.3. Using the data, accurate waiting times for baggage drop and baggage reclaim times are extracted. Furthermore, the information might shed light on processes currently not incorporated into the model which should be. Alterations to be made to the model is discussed in the next Section model improvement.

3.3.3. Model Improvement

Using iterative loops of looking at increasingly more complex case studies, the sources for model error are examined and tried to mitigate. The data analysis provides the reference data which can be used for calibration and validation. It is here where it is to be investigated how tuning is to be done. Tuning will be done based on three performance indicators, being security checkpoint occupancy (number of passenger present in the system), security checkpoint time (the time required for a passenger to enter and exit the security checkpoint), and throughput (the number of passengers entering the system per time unit, usually per minute). These three parameters are needed since although an accurate occupancy can be achieved, the security checkpoint time might be far to small or too large. Hence, these three parameters combined describe the checkpoint performance in full.

Cases considered are the security checkpoint operation during the morning bank of April 17 2018 and the morning bank of March 12 2018 as will be described in Section 4.1. These cases are chose as they have favourable characteristics. April 17 has a homogenous population, while March 12 has a heterogeneous population, while data-pollution due to earlier flights banks is eliminated.

As a starting point, it is not uncommon to look at the modelled processes and use relatively simple representations - in AATOM through waiting times - to initiate the model, derived from the experimental data. Simple does not mean only a frequently used distribution is chosen such as a Normal distribution. It also

applies to the foundation, interpretation and data-fusion leading up the resulting distribution chosen. Put differently, a normal distribution can be the result of a an elaborate investigation and is therefore not simple.

This setting of parameters will yield certain outcomes of the system which might produce (partially) valid outcomes. A more in-depth look can be taken at the available data to get a better mathematical representation of the processes described to get increasingly valid outcomes. The validity of the outcomes are assessed through comparing the experimental performance results with the simulated performance results. Through iterative loops of tuning the parameters - calibration - it is tried to get outcomes which represent a reality to a sufficient degree; validation [42].

3.3.4. From Methodology To Execution

In the next Chapter 4, the research question is tried to be answered. This is done using the theory, knowledge and tools presented in the literature study in Chapter 2, guided by the aforementioned sub-questions using the proposed methodology. Chapter 4 starts with describing the data in Section 4.1 followed by how it is analysed in Section 4.2. Next, Section 4.3 delves into how the analysed data and insights can be transferred to the simulator AATOM. A logical next step is analysing how this transfer of information is (to be) done and the subsequent lessons learned and increased understanding found. This will be investigated in Section 4.4. A conclusion will be given in Chapter 5..

4

Agent-Based Model Calibration & Validation

As mentioned in previous Chapters, a model is made and updated to represent the actual security checkpoint operations to a sufficient degree . Although the relevant processes might be conceptually included in the model, it does not mean it already resembles the real-life process. The purpose of calibration is trying to let the relevant processes resemble the real-life processes and is done during calibration. Klügl formally defined it as *'parameters have to be set in a way that a structurally correct model produces a valid outcome'* [28]. In the subsequent Sections in this Chapter, the tasks to be completed are discussed how to get a calibrated and validated model.

4.1. Data Acquisition

To be able to populate the model with data to represent reality, data needs to be gathered of the actual processes under investigation. This is done through meticulously observing and recording data of real-life operations, done at Rotterdam The Hague Airport. The analyses of the gathered data is the topic of Section 4.2. It should be noted that the data gathered will be more detailed than allowed for by the existing AATOM. Hence, a modification is implemented.

An addition to be made to AATOM is the inclusion of heterogeneous population. To this end, categories of passengers must be distinguished. According to domain experts from RTHA six different passenger could be identified. The distinction between the groups are made on the basis of expected time required to go through the processes of the security checkpoint, while being able to visually categorise them. The following list describes the six types and their respective characteristics. It should be noted that although guidelines are formulated, still, human inferences are made and ambiguous cases exist. Using the Fuzzy-logic principle [20], a passenger can be attributed to multiple categories when deemed applicable. For instance, a Businessman/woman can also be Young.

1. Business:
    - Business laptop, item roll suitcase, formal garments (jackets/suits)
2. Senior:
    - 60+
3. Family:
    - Parent with child(ren)< 18 year
4. Young:
    - 18-35 year
5. Handicapped:
    - Every form of being observable disabled (wheelchair, crutches, visual impairment, etc.)

6. Regular:
    - None of the above

The operation is analysed through security footage analysis. 160 hours are spent in analysing 14.5 hours of security footage of the security checkpoint. This 14.5 hours comprise nine flight-banks on eight days between March 10th and April 17th, as can be seen in Tables 4.1, 4.2, 4.3, and 4.4. As the data is recorded on different days, it is tried to establish upfront to what degree similar or different conditions are present, which might introduce errors in the gathered data. Categorisation is primarily based on the expected passenger type to encounter, where business banks, leisure banks, and a combination are identified.

Table 4.1: Case 1: gathered at RTHA in 2018

Case 1	**Destination**	**Departure**	**Flight**
March 10: 14:30 - 15:57	Fara	15:50	HV6093
	Malaga	16:25	HV5023
	Alicante	16:45	HV5053
March 31: 14:29 - 15:28	Fara	15:50	HV6093
	Malaga	16:25	HV5023
	Alicante	16:45	HV5053
April 7: 17:02 - 18:36	Pisa	18:25	HV6421
	Malaga	18:55	HV5021
	Alicante	19:30	HV5053

Table 4.2: Case 2 gathered at RTHA in 2018

Case 2	**Destination**	**Departure**	**Flight**
April 17: 8:00 - 10:04 & 13:00 - 14:18	London	09:55	BA4452
	London	10:55	BA4454
	London	13:55	BA4476
	London	15:45	BA4456

Table 4.3: Case 3 gathered at RTHA in 2018

Case 3	**Destination**	**Departure**	**Flight**
March 12: 05:13 - 06:14	Alicante	07:00	HV5053
	London	07:05	BA4450
	Innsbruck	07:10	HV6833
	Budapest	07:15	HV6771
	Valencia	07:20	HV6441
	Malaga	07:30	HV5021

Table 4.4: Case 4 gathered at RTHA in 2018

Case 4	**Destination**	**Departure**	**Flight**
April 15: 5:01 - 06:30	Venice	06:55	HV6493
	Malaga	07:00	HV5021
	Ibiza	07:05	HV5689
	Alicante	07:40	HV5053
	Faro	07:50	HV6091

The maximum temperatures during these days range from 0 to 14 degrees Celsius. This is important since passengers are more likely to be wearing more clothing with colder temperatures, which is expected to result in longer processing times as more baggage has to be handled.

Also, the time of the day of the flight-bank is expected to be of influence on the processes. Possible drowsiness during the early hours of the day or tiredness near the end of the day can be an influencing factor on the time it takes to process the passengers, both due to the passengers, as well as personnel. As such, the nine available flight banks are grouped into 4 cases which exhibit similar flight bank characteristics. Case 1 and 2 consist of multiple flight banks while case 3 and 4 are single flight banks.

Furthermore - as mentioned before - flight-bank composition is of importance. Different destinations attract different kind of passengers. Flights to London are mostly used by business passengers. Destinations such as Innsbruck and Alicante are both primarily leisure destinations. It should be observed that the type of leisure differs, as Innsbruck is a known skiing destination, while Alicante is more geared towards enjoying the favourable weather conditions. This might be important as a different luggage composition is reasonable to be expected.

Finally, the composition of the security checkpoint is to be considered. The number of and destinations of the flight-bank in combination with the number of open security lanes result in both a certain arrival rate of certain passenger types, as well as a certain capacity to process these passengers. This interplay of arrival rate and capacity will be of influence in queue forming, and if it does, how long the queue becomes. The impact this might have, is that a long queue could possibly result in relaxation of protocols on part of the security personnel. Furthermore, the passenger might be inclined to perform their tasks above normal speed as one wants to clear the security checkpoint as soon as possible. This could influence the recorded process times and should be recognised.

The aforementioned paragraphs describe the care that should be taken when analysing the data, which will be the subject of Section 4.2. Yet, despite these imperfect conditions, an attempt is made to gather data as accurately as possible, as no alternative to improve the quality of the data is available.

The data gathered per passenger is done using a template, ensuring it is standardised, equal for all passengers. The list of entries can be seen below.

1. Passenger type(s) (Business, Senior, Family, Young, Handicapped, Regular)
2. Prepared or not (Prepared if a passenger is not distracted during the security checkpoint process, as well as having knowledge of the steps to be taken)
3. Baggage drop start time (unloading starts)
4. Number of baggage boxes used to feed into the x-ray scanner
5. Baggage drop stop time (unloading finished)
6. Walk-through metal detector (WTMD) time (First passage through WTMD of the passenger under investigation)
7. Additional manual screening after WTMD start time (interaction with checkpoint officer starts)
8. Additional manual screening after WTMD stop time (interaction with checkpoint officer finished)
9. Explosive trace detection (ETD) start time (interaction with checkpoint officer starts)
10. ETD stop time (interaction with checkpoint officer finished)
11. Baggage reclaim start time (reclaim starts)
12. Baggage reclaim stop time (reclaim finished)
13. Additional baggage screening start (interaction with checkpoint officer starts)
14. Additional baggage screening stop (interaction with checkpoint officer finished)
15. Comment section (used to register observations not allowed for in the framework)

4.2. Data Analysis

The gathered data is now to be subjected to data analysis. To this end, the Matlab environment is used. The sixteen data-entries as described in the previous section are loaded into Matlab. Using scripts automatically load and analyse the data selected. Data selection can be done through analysing the days individually or all the days combined. Also, the passenger type can be investigated separately, as well as combined. These options are available for the relevant (sub)-processes which can be distilled from the information recorded in the template, as described in the previous Section 4.1.These sub-processes are:

1. Baggage drop time
2. Time between finishing the baggage drop to passing the walk-through metal detector for the first time
3. Additional walk-through metal detector passes time
4. Time between finishing the baggage drop and starting the baggage reclaim
5. Walk-through metal detector check time
6. Explosive trace detection check time
7. Baggage reclaim time
8. Baggage-check time
9. Total time to clear the security lane of the security checkpoint. Time from start baggage drop to finishing baggage reclaim

With the dataset and processes of interest chosen, the Matlab scripts load the relevant data. Based on this data, distributions are fitted using the built-in 'fitdist' command. Here, the discrete data found at RTHA are now converted into a continuous probability function. Using a maximum likelihood estimation, the parameters of the considered distribution are set. The fitted functions are subjected to a 'goodness of fit' test to asses how well the continuous function describes the gathered data. To this end, the Kolmogorov-Smirnov test is used. Through comparing the fitted - experimental - cumulative distribution function with the cumulative distribution of the data gathered at RTHA, the difference between the two can be found. The maximum difference is a metric for how well the distribution fits. If the test statistic $p > 0.05$, it is assumed the samples are drawn form the same distribution [21], [40]. Larger values of p therefore indicate a better fit to the available data. This does not mean the best fitting distribution should also be implemented in the agent-based Model, as overfitting might occur [18]. As a particular distribution resembles the data too closely, generalisation might be lost. After all, now, the model is tuned as closely as possible to the experimental data, which might only be a special case of the generalised process under investigation.

In order to mitigate the overfitting risk, a directive is used in choosing a fitted distribution resembling the sub-processes given a chosen data-set. This directive is based on previous literature [37], [45] and an assessment of the Smirnov-Kolmogorov values as passing the 'goodness-of-fit' metric is still required. The findings in previous literature can be seen in the next Table 4.5.

Table 4.5: Distributions from literature per process

Process	Distribution [45]	**a**	**b**	λ	Distribution [37]	**a**	**b**	λ
dropBaggage	Gamma	14.38	2.29	-	Gamma	5.22	0.15	-
WTMD	Log-normal	3.31	0.30	-	Log-normal	3.62	0.6	-
X-ray	Exponential	-	-	0.05	Exponential	-	-	0.066
baggageReclaim	Gamma	25.67	2.18	-	Exponential	-	-	0.14

A subset of the available distribution types in Matlab which can be used as input is chosen. Using 'brute-force', this subset of distributions is tried to be fitted to the available data using the 'distfit' command. This set can be seen in the list below.

1. Birnbaum-Saunders
2. Burr
3. Exponential
4. Extreme Value
5. Gamma
6. Generalised Extreme Value
7. Generalised Pareto

8. Inverse Gaussian
9. Logistic
10. Log-logistic
11. Log-normal
12. Nakagami
13. Normal
14. Poisson
15. Rayleigh
16. Rician
17. Stable
18. t-Location Scale
19. Weibull

As described in the previous paragraph in combination with Table 4.5, of first choice are the Gamma, Log-normal, and Exponential distribution. If these distributions prove insufficient, the second choice distributions constitute the Burr (overlaps, or has as a limiting case, many commonly used distributions such as Gamma, Log-normal, Log-logistic, Bell-shaped, and J-shaped beta distributions), Poisson (appropriate for applications that involve counting the number of times a random event occurs in a given amount of time), Log-logistic (survival analysis to model events that experience an initial rate increase, followed by a rate decrease), or Weibull/Extreme Value (reliability and lifetime modelling. The Weibull distribution is more flexible than the exponential for these purposes/closely related to the Weibull) [48]. This set is compiled based on previous use-cases of the distributions [34]. If these distributions prove to be insufficient as well, a closer look is taken at the remaining distributions and a case-to-case explanation for the choice made is given. A closer look is taken at the data used for this distribution fitting.

4.2.1. Data Split & Merge

Data might be merged. For instance, it can be useful to test data for similar processes, recorded separately, stem from the same distribution (homogeneous). Larger data-sets allow for achieving a better statistical power, resulting in more confidence in the results found. A risk of data-merging is that some behavioural patterns might get lost. To this end, the Kruskal-Wallis test could be employed, as it allows for datasets larger than two. The Mann-Whitney test could not be used for all cases [40], as this test allows only two datasets.Therefore, for p-value interpretation purposes, it is chosen to use the Kruskal-Wallis for all cases, despite losing some power in certain cases. When the null hypothesis is not rejected - $p > 0.05$ - data can be merged. The tables for the tests per case can be found below in Tables 4.6, 4.7, 4.8, 4.9, and 4.10. It should be observed that not all passengers were present during the different cases. Hence, only the relevant passenger types are listed. Furthermore, the Regular passengers are not necessarily uniquely Regular, as the Fuzzy logic allows for passengers being classified into multiple passenger types. In practice, it is found that due to the broad definition, only Regular is used in combination with one of the remaining five categories.

Table 4.6: Data Merge Passenger Types Case 1 March 31 14:29 - 15:28 & April 7 17:02 - 18:36 2018: 2 Regular lanes operational

Case 1	**p-value baggageDrop**	**p-value baggageReclaim**	**p-value securityCheckpoint**
Senior (49 pax)	0.0024	0.2141	0.0042
Family (64 pax)	0.0004	0.0434	0.0006
Young (44 pax)	0.0319	0.2150	0.1573
Regular (119 pax)	0.0691	0.4281	0.8462
AllPassengers (253 pax)	0.0069	0.7023	0.8125

Table 4.7: Data Merge Per Passenger Type Case 1B March 31 14:29 - 15:28 & April 7 17:02 - 18:36, 2018: 2 Regular lanes operational

Case 1B	**p-value baggageDrop**	**p-value baggageReclaim**	**p-value securityCheckpoint**
Young vs Regular March 31 (93 pax)	0.0488	0.5579	0.0775
Young vs Regular April 7 (100 pax)	0.9969	0.0890	0.3072
Senior vs Young March 31 (37 pax)	0.0072	0.5635	0.0023
Senior vs Young April 7 (57 pax)	0.1692	0.2665	0.6316

Table 4.8: Data Merge Per Passenger Type Case 2 April 17 8:34 - 10:04 & 13:00 - 14:18, 2018: 1 Regular lane operational

Case 2	p-value baggageDrop	p-value baggageReclaim	p-value securityCheckpoint
Business (106 pax)	0.0797	0.5212	0.2761
Young (7* pax)	0.4795	0.1573	0.1573
Regular (76 pax)	0.7734	0.7010	0.8572
AllPassengers (143 pax)	0.5844	0.8988	0.9822

Table 4.9: Data Merge Per Passenger Type Case 2B April 17 8:34 - 10:04 & 13:00 - 14:18, 2018: 2 Regular lanes operational

Case 2B	p-value baggageDrop	p-value baggageReclaim	p-value securityCheckpoint
Business vs Regular April 17 Morning (146 pax)	0.4591	0.4876	0.7016
Business vs Regular April 17 Afternoon (36 pax)	0.2952	0.5260	0.4098

Table 4.10: Data Merge Per Passenger Type Combined: All flights as used in the 4 cases combined

Case Combined	p-value baggageDrop	p-value baggageReclaim	p-value securityCheckpoint
Business (106)	0.0797	0.5212	0.2761
Senior (106 pax)	0.0109	0.1540	0.0026
Family (83 pax)	0.0007	0.1211	0.0006
Young (94 pax)	0.2059	0.0090	0.0011
Regular (303 pax)	0.4433	0.0514	0.1725
MiVa (8* pax)	0.1797	0.2453	0.2453
AllPassengers (609 pax)	0.0363	0.0008	0.0000

Conclusions that could be drawn from these tables is that the creation of a heterogeneous population is essential when looking at a larger macro level. In Table 4.10, it can be seen that stratification based on passenger type increases the probability the process investigated could be drawn from an underlying fitted distribution. The test executed for different passenger types on the same day show a less conclusive result, as can bee seen in Tables 4.7 and 4.9. This as in some cases the test suggest being drawn from the same underlying distribution is very likely to unlikely. This might be due to the fact that conditions within the day not accounted for have a larger influence on the process times than passenger type. The existence of differences per day is supported when looking at Table 4.10. Here, it shows that the processes investigated behave differently throughout the different cases, as the p-values decrease sharply when compared to Tables 4.6 and 4.8 (where data from different days under relative equal loading conditions is merged). Hence, although depending on the conditions, different passenger types and its implementation into AATOM can assist in the calibration and validation efforts.

4.3. Data Implementation

Information extracted from the database populated with observations is used to calibrate to the agent-based model AATOM. For an in-depth analysis of the model, please consult Appendix A. However, a condensed description will be given next with the focus on use in this project rather than the more descriptive explanation given in the literature review, Section 2.

As described in the AATOM documentation [26] and the literature review in Section 2.2.3, the security checkpoint is one of ten areas accessible to agents. In these areas, physical objects are placed to form the environment. This includes the queue separators used to implement the snaking of the queues, as well as the layout and devices used in the security lanes. These sensors forming the lanes consists of x-ray machines, WTMD's, and ETD's. Also, places for operator-agents are reserved. These parts together form the environment of the security checkpoint in the model.

Three agent types operate in the environment. Passenger agents represent the passengers at an airport. Furthermore, operator agents are modelled. These act as employees working at the airport. At the security checkpoint, these operators represent among other things employees tasked with the boarding pass check, operating the x-ray machines, and performing the ETD-check. Thus, both passenger and employee agents are humanoid representations in the form of agents and are henceforth described as human agents. Finally, orchestration agents can be modelled. Either in the form of a human or non-human entity, are capable of steering operator and/or passenger agents and adjust the environment to achieve specific goals, such as going to the security checkpoint and afterwards towards the gate.

The human agents are composed of a three-layer structure and can be viewed as the 'human brain', resulting in the agent's behaviour (which is tried to resemble human behaviour). The strategic layer determines the goals and updates the belief module. Furthermore, reasoning is done in this layer and includes performing analysis resulting in planning and decision-making. The middle layer - the tactical layer - is tasked with actuation of activities. This includes interpretation of observations and route navigation. The third and final layer - the operational layer - is responsible for interaction with the environment and interaction with other agents. Input is sensory data from the agents sensors. Output are the actions, defined by the input, going through the tactical, strategic, and tactical layer, for eventual execution by the operational layer.

Before concluding the AATOM description, it is essential to get a better understanding in how the implementation of the security checkpoint at level II aggregation is done. This will expose the options, tools, and parameters available for manipulation and calibration. It can be read in [24] that the processes within the security lane - both the sensors, as well as interactions with operator agents - is modelled through a predetermined waiting time. This predetermined waiting time is a direct result of the action being undertaken. For instance a waiting time resembles the time required to perform the baggage drop. This baggage drop time can be supplemented by an interaction with a baggage drop security operator, which represents the additional assistance given to assure the baggage is dropped off correctly. Waiting times are used as only the observed behaviour is tried to be represented. Put differently, the cognitive processes leading up to behaviour observed at the processes within the security lane are not modelled. Merely the consequences thereof - the time it takes to go through a process - is registered and used in the model.

These waiting times can be set differently for different processes, for different passenger types. The processes chosen are those from the perspective of the passenger whilst having a substantial influence of execution on the task. During the x-ray check or WTMD check, the security officer has a larger influence compared to the passenger. As the perspective from the passenger is chose, the emphasis and allocation of resources is aimed at processes where the passenger has the largest influence. This view is updated if deemed necessary. The discrete waiting times per passenger type are listed below:

1. Baggage drop time
2. Baggage reclaim time

Finally, as described in the literature review in Section 3 a hybrid approach is used, where calibration and validation is performed using the (classical) generalised framework in light of a data-driven approach. The initial domain/expert knowledge based model is taken as a starting point which is supplemented by insights gained based on the data, ranging from statistical analysis to 'comment section' analysis. It may also include

alterations to be made to the model. Hence, when found necessary, the investigation stretches further than only input distribution manipulation.

4.4. Model Analysis

Given an unaltered simulator structure, not only input parameters have influence on the outcome of the simulations. Also, stochastic elements within the simulator yield different results per run. These stochastic elements do however start to stabilise if the number of simulations are increased, according to the central limit theory [40]. It is therefore vital to asses what number of replications is required. This will be discussed in Section 4.4.1. These results are used for further model refinement.

4.4.1. Confidence Interval and Number of Replications

Results per simulation run differ due to stochastic effects within the model [29]. As each simulation run is expected to yield (slightly) different results, a number of replications is to be defined to give to a certain degree accurate results while also limiting the computational efforts. The number of runs is established through confidence interval analysis. 1000 runs with constant input parameters but with the seeds enabled are executed (each replications has a different seed). The seeds are used to draw values from the baggage drop and baggage reclaim distribution. Furthermore, the seeds influence when the passengers are spawned in the simulator. Hence, for changing seeds, different values are drawn and results in a different simulation while the input parameters remain constant. It should be observed that under the same seeds, the model will behave identical. The confidence interval is analysed for the first case under investigated. This case will be discussed in Section 4.4.2 and determines the number of replications required for all simulations to comes. Incrementally simulations are added to compute the confidence interval. The metric used is the average security checkpoint process time in seconds. This model output is chosen since the security checkpoint encapsulates all seeds in the simulator, making it suitable for replication determination. Seconds are chosen as this is the output unit for the security checkpoint time. After 52 runs, it is found that the half-width confidence interval around the mean is less than one percent of the mean. It should be observed that Fisher's 95% criterion is used. Given the specification of the server available for analysis - having 128 cores - the number of runs is more than doubled to 128 runs per investigated case, as this scales well to the physical characteristics of the server and additional precision is gained at relatively little cost. The confidence interval progression with increasing sample size is given in Figure 4.1. It can be seen that at 128 simulations, the curve has tapered off significantly.

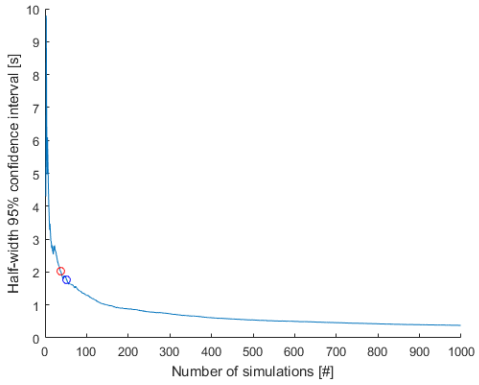


Figure 4.1: Half-width confidence interval progression with increasing simulations runs. The red dot is a half-width confidence interval around the mean of 2 seconds (46 replications), while the blue dot (52 replications) is a half-width confidence interval around the mean within 1% of the sample mean

4.4.2. AATOM: A Baseline Case

When first analysing the model, a test case as simple as possible is sought after. Ironically, while the model is adapted to allow the use of a heterogeneous population, a test case with an as homogeneous population as possible is preferred, as this will reduce the chance of differences being the result of a heterogeneous population, but rather model design or input parameter setting. Put differently, the behaviour of the model is investigated in an as baseline case as possible. The test case most suitable is Case 2 in the morning, Table 4.2. This since it is a relatively isolated flight schedule where the latest flight before departed at 08:00 and the next

flight at 13:55, as can be seen in Table 4.11. Furthermore, given the time in the morning, being on a Tuesday and its destination London, a relative homogeneous - primarily business - population is observed.

Table 4.11: Relatively isolated morning flights April 17 to London.

Case 2: Morning	**Destination**	**Departure**	**Flight**
Last in front	Malaga	08:00	HV5021
April 17 8:00 - 10:04	London	09:55	BA4452
	London	10:50	BA4454
First behind	London	13:55	BA4476

First, a closer look needs to be taken at the experimental data gathered at RTHA for this flight bank. A total of 102 passengers are tracked between 8:00 and 10:02 for this flight bank which went through the single operational security lane in the security checkpoint. Since this flight bank is so isolated, it is reasonable to assume the majority of the recorded passengers take one of both flights. It does however not eliminate inferential effects from flight-banks earlier and late and independence is not guaranteed. Still, it is deemed the best flight bank - given the available data - as it is an as simple case as possible while still being relatively isolated. It is interesting to observe how these passenger have gone through the security checkpoint through time.

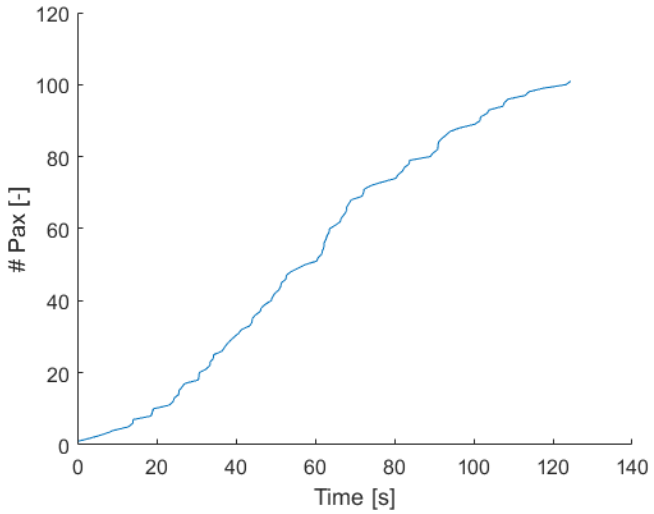


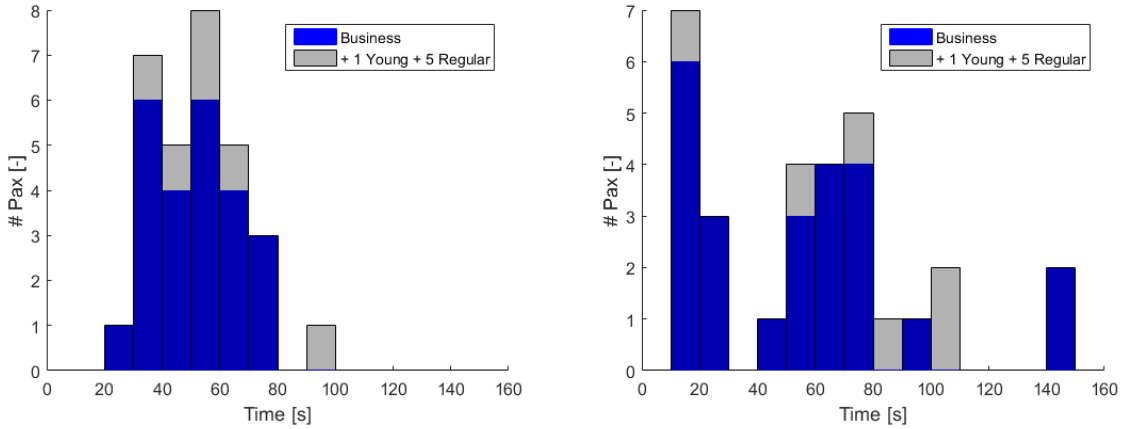
Figure 4.2: Throughput passenger April 17 08:00 - 10:04 at the security checkpoint at RTHA

As expected, a monotonically increasing graph is seen Figure in 4.2. What can be observed as well is that the gradient - thus the throughput which is number of passengers per time window - increases from 0 (08:00) to 30 (08:30) minutes as more passengers arrive at the checkpoint. Here, the gradient stabilises as a queue is formed and saturation of the system is achieved. A dip can be seen at 50 (08:50) minutes. At 60 (09:00) minutes a sharp increase can be observed before it tapers near the end of the observation window, as the system no longer is saturated due to a smaller influx of arriving passengers. Of special interest is the time frame between 30 and 50 minutes. Here stable, saturated operations is achieved. Because the gradient does not suffer from much noise, as well having a constant , largest gradient. In this 20 minute window, 30 passengers are processed, resulting in $1.36\frac{pax}{min}$. At 60 minutes a larger gradient can be observed. It is of only of short duration. Furthermore, it is being preceded by a period of a very small gradient where passengers were given time to prepare themselves more thoroughly for the tasks to be completed, resulting in higher performance. Hence, it is an overestimation of stable maximum performance that can reliably be attained. This is supported by the notion it tapers off quite quickly to levels similar or lower than stable maximum performance after 60 minutes.

With the general performance of the security checkpoint known, a more thorough look has to be taken at the data gathered. Firstly, passenger composition is of interest. These 102 passengers could be categorised into three of the available six types. These are Young, Regular and Business. Since passengers can be attributed to

multiple types - as described in Section 4.1 - the combined types recorded are greater than actual passengers which went through the system. 4 young passengers were recorded, 57 Regular, and 89 Business. It should be observed that only 9 Regular passengers were 'just' Regular. Thus 48 passengers were both Business and Regular. In total, 28 passengers had to go through the metal detector twice, with taking off their shoes accounting for 25 times (which have to go through the x-ray machine if the WTMD goes off). The remaining three reasons were a belt, a wallet, and a watch. 5 additional screenings due to going off of the WTMD were performed. It is interesting to observe these are all preceded by having to go through the WTMD twice. An ETD check was performed a total of 14 times. Finally, 7 additional baggage searches were executed for further investigation of items being brought along. In the comment section from the recorded data no noteworthy data can be reported. It should be reiterated the comments sections allows for recording data not allowed for by the template.

First, we tried to replicate the simulator behaviour during steady maximum performance between 30 and 50 minutes. This is calibration. To this end, the data analysis is done on the 30 passengers observed in this window. This applies to both passenger type composition and distribution fitting for the baggage drop and baggage reclaim. The experimental data describes a saturated system. Therefore, in the simulation, also the performance during system saturation is investigated. Again, lets take a closer look at the data. 24 Business passengers are recorded, of which 19 were categorised as also being Regular. Furthermore, 5 Regular passengers were tracked and 1 Young. 7 double WTMD passes have been performed, 1 additional screening as a result of passing the WTMD, as well as 5 ETD checks. No additional baggage searches were performed.



(a) Baggage drop

(b) Baggage reclaim

Figure 4.3: Histogram baggage drop April 17, morning

As can be seen from the Histograms 4.3, the differences between Business only and the complete dataset for the baggage drop is small. The baggage reclaim is less conclusive, as the spread is larger, irrespective of looking only at Business passengers or all. A closer look is taken at extreme values for baggage reclaim, being either very fast (20 seconds) or relatively slow (140 seconds). The comment section of this April 17 morning bank did not provide additional information about the possible reasons for either being fast or slow. The number of boxes, groupsize, or being experienced or not did not differ from the other passengers. Hence, the recorded data does not give us a simple answer for the differences observed.

Due to the limited data available during this window a homogeneous population with the data of a heterogeneous populations is created. Hence, the passenger simulated is primarily business, with a hint of regular and young. Distributions are fitted to the available data, as described in Section 4.2. Using the choice directive describing which distributions are preffered, described in this same Section, the following distributions with accompanying parameter settings are chosen. This can be seen in Table 4.12.
The distributions from Table 4.12 are implemented into AATOM. 128 replications are done where as input 102 agents are spawned. More passengers than experimentally found are used to be able to compare stable saturated performance. Firstly a queue has to be formed. Furthermore, due to random seeds, some simulations run longer than others. As to be sure all the simulations can be compared using the same window,

Table 4.12: Selected input distributions based on data gathered April 17, morning during security checkpoint saturation

	Distribution Type	**Kolmogorov-Smirnov test**	Rank
Baggage drop	Weibull: Scale: 58.1001 Shape: 3.6867	0.9057	1 out of 19
Baggage reclaim	Normal: Mean: 58.1333 Standard deviation: 37.3231	0.6744	1 out of 19

not the entire simulation run is compared, but rather a smaller identical time-window - snippet - is cut from each run and compared. Therefore, also, a snippet between the beginning and the end is compared with the experimental data, found under similar circumstances. The replications are done as results might differ due to random generators within the model. Again, it is tried to replicate saturated performance as observed at RTHA in the simulator. Each line in Figure 4.6 represent one of the 128 replications. The thick yellow line is the experimental data. As mentioned in Chapter 3, tuning will be done based on three performance indicators, being security filter occupancy (number of passenger present in the system), security checkpoint time (the time required for a passenger to enter and exit the security checkpoint), and throughput (the number of passengers entering the system per time unit, usually minutes). These three parameters are needed since although an accurate occupancy can be achieved, the security checkpoint time might be far to small or too large. Hence, these three parameters describe the checkpoint performance in full. For now, only the throughput is considered as the influencing factors on this model output is tried to be found. Also, the average runtime is to be investigated. With a 95% confidence interval, the number of passengers spawned per simulation are [90.3402,92.9880] with a mean of 91.6640 passengers. This occurs around 45 minutes. The 95% confidence interval for the throughput during saturation - measuring between 10 minutes and end of simulation - is [2.0901,2.1219] with a mean of 2.1060 passengers per minute. Here, the end of simulation can be used as no tapering can be seen, while saturation of the system takes time as the arrival of passengers needs to be higher than the service rate. This saturation point is chosen to be 10 minutes.

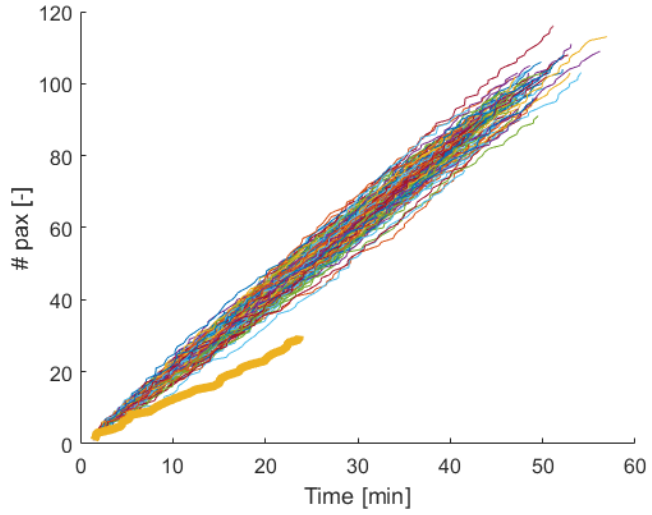
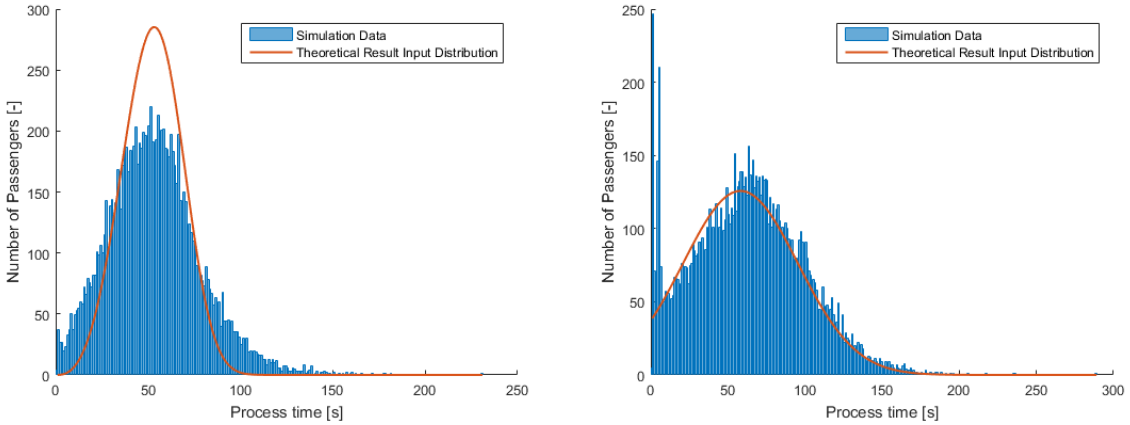


Figure 4.4: Throughput simulated passenger during saturation April 17 08:30 - 08:50 at the security checkpoint at RTHA. 128 Replications. The thick yellow line is the experimental data.

It can be concluded that the simulation over-performs with respect to the experimental data, which was found to operate at 1.36 passengers per minute. Currently, no additional checks are implemented into the model. As these checks take up time, it is expected to lower the performance of the model. Another possibility might be 'system occupation'. This is the number of passengers being in the security lane between baggage drop en baggage reclaim. In the model, 3 slots are available for both processes. Hence, a total maximum of 6 passengers can be active in a security lane. In practice, this might be different. During the experiment, the mean occupation is found to be 3.8918. During simulation, the mean was 5.7343 with a confidence interval

of [5.7168,5.7519]. This is a significant difference in occupancy.

The change of occupancy in itself does not paint a complete picture however. Although more people are in the system, they are not necessarily actively performing a task, as one might have to wait for a place to become vacant. Only then, moving to the next process becomes a possibility. Hence, occupancy is required to be examined with the process times as well. Waiting might inflate the realised process time with respect to the defined input distributions. Therefore, a (graphical) analysis is to be conducted. Observe that a resolution for the histogams (bin width) of 1 second is chosen, as this is equal to experimental data.



(a) Baggage drop

(b) Baggage reclaim

Figure 4.5: Process duration 128 simulations versus input probability density functions during saturation. The red line is the theoretical input found after distribution fitting on the experimental data

The general shapes of both processes correspond to the input distributions provided as the centre of the peaks are comparable. Baggage drop displays a broader, less high peak compared to the theoretical value (more platykurtic). The peaks do occur at the same time. This behaviour can be explained when looking at the mean process times. Baggage drop takes more time than baggage reclaim (63.3548 versus 53.2278 seconds). Hence, while the baggage drop might be concluded, an agent has to wait for the baggage reclaim spot to become available. Hence, the input distribution is less well replicated when a looking at the realised distribution after simulating, compared to baggage drop. This waiting cannot be mitigated by the large spike near the origin where many passengers take a short time to reclaim. Baggage reclaim does show similar kurtosis, comparable to the input distribution but produces outliers near the origin. This can be attributed to the fitted Weibull function as described in Table 4.12. The left tail provides a non-zero probability for values below 0. In the simulator, these values smaller than zero are given a positive non-zero value close to 1. The cumulative distribution function indicates the left tail for values smaller than 1 second account for 6.24% of the total values. With 11733 simulated agents in 128 replications, this amounts to 732 passengers, which explains the outliers near the origin.

With the differences known concerning the occurrence of additional checks (or the lack thereof), the occupancy (simulated higher than during the experiment) and the reproduction of input distributions (baggage drop influenced by baggage reclaim while the input distributions introduces outliers near the origin), a closer look has to be taken at how well the simulator under these circumstances and known caveats performs with respect to the security checkpoint time. In similar fashion when determining the baggage drop and baggage reclaim input distributions, 30 available passengers are used to construct a theoretical distribution, to be compared to the simulated results. When taking into account the distribution selection directive and testing the goodness-of-fit with both the Kolmogorov-Smirnov and Chi-Squared test (summing to find the highest score total fit score as Kolmogorov-Smirnov alone produced an inconlusive result due to multiple distributions being close to 1), a Normal distribution is selected to describe the experimental data, as can be seen in Table 4.13. In similar fashion, the theoretical output can be compared with the empirical output. This can be seen in Figure 4.6.

Table 4.13: Selected output distributions based on data gathered April 17, morning during security checkpoint saturation

	Distribution Type	Kolmogorov-Smirnov Test	Chi-Squared Test	Rank
Security checkpoint	Normal: Mean: 145.4000 Standard deviation: 42.8177	0.9895	0.9523	3 out of 19

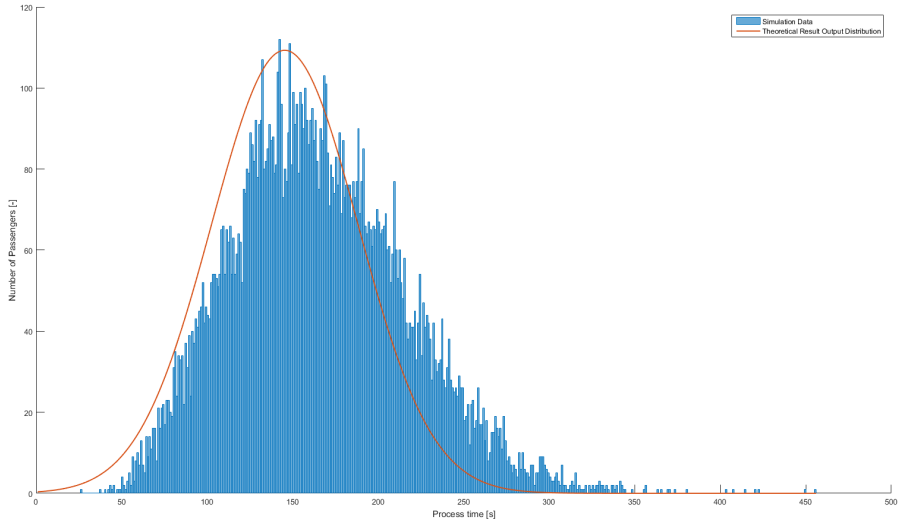


Figure 4.6: Process duration 128 simulations versus output probability density functions during saturation

When visually inspecting the graph, the general shape is comparable. What is striking to see is that the mean of the simulation data is shifted to the right (higher time) compared to the theoretical distribution. This, despite the fact that the simulation has a higher occupation and has no implementation of double WTMD passes or a patdown as a result of not passing the WTMD. Mathematically assesing the goodness-of-fit using Kolmogorov-Smirnov, Chi-Squared or other tests becomes difficult when the sample size increases. This, since even trivial deviations become inflated and the error parameter becomes large, resulting in (not representative) low p-values. To circumvent this problem, a new function is fitted to the simulation data. The distribution fitted is of the same type as the experimental data, meaning a Weibull and two Normal distributions are used respectively. Comparison of these fitted distributions using experimental and simulated results are used to asses how well the experiment and simulation compare. The observations during the graphical analysis are supported by the values found in the Table 4.14.

A further investigation needs to be conducted to try and understand how the throughput of the security checkpoint results from all subprocesses encountered. Of special interest are the occupation and security checkpoint time. To asses possible relations, a correlation matrix is constructed. Correlations can reveal if sub-process are related and timing data can be merged and construct more realistic baggage drop and baggage reclaim input distributions. Correlation provides a linearised non-parametric relation between two model outputs of interest and is a value between -1 and 1. A positive correlation indicates that if one parameter is larger/smaller, the other becomes larger/smaller as well. Negative correlation means movement in the opposite direction. If one parameter becomes larger/smaller, the other parameter becomes smaller/larger. Furthermore, the closer the value to either 1 or -1, the stronger the effect (and closer to zero, the smaller) [40]. Only values are given where the significance threshold the $p \leq 0.05$ was satisfied. If the correlation coefficient proved insignificant, the value is nullified. The results can be found in Table 4.15. Note that baggage check is not included, as no baggage checks were conducted for this sample. Observe that all the available measured data is included in the correlation study and not all data is used or translated to the simulator. This is done to be able to identify if particular processes (not directly included in the simulator) require more attention. Intuitive results are found when looking at the correlation between baggage drop, the baggage reclaim and the security checkpoint. High positive values can be found. After all, longer time dropping and collecting luggage is expected to increase the total security checkpoint time.

Table 4.14: Distribution parameters and properties: mean and standard deviation

	Distribution	Mean	Standard deviation
Baggage drop Experimental	Weibull: Scale: 58.1001 Shape: 3.6867	52.4223	15.8308
Baggage drop Simulated	Weibull: Scale: 59.9096 Shape: 2.2602	53.0625	24.8520
Baggage reclaim Experimental	Normal: Mean: 58.1333 Standard deviation: 37.2331	58.1333	37.3231
Baggage reclaim Simulated	Normal: Mean: 63.3548 Standard deviation: 35.4709	63.3548	35.4709
Security checkpoint Experimental	Normal: Mean: 145.4000 Standard deviation: 42.8177	145.4000	42.8177
Security checkpoint Simulated	Normal: Mean: 165.5502 Standard deviation: 51.4162	165.5502	51.4162

Two large correlations ($\rho \geq 0.50$) are found in the matrix [21], [30]. Baggage reclaim has a large positive correlation with the security checkpoint time (0.8735). Baggage reclaim is of larger influence than baggage drop as reclaim is a process with a higher average time. Hence, passengers have to wait for a spot to become empty at baggage reclaim. Experience has a large negative correlation with the number of times going throuhg the WTMD twice (−0.5100) . Noteworthy is the observation that occupation has no statistically relevant correlation with any of the investigated parameters, and hence not on the security checkpiont time. The checkpoint time is influenced by the number of X-ray boxes (0.3773) and experience (−0.4037). These are medium effects, as the value is between ($0.3 \leq \rho < 0.5$) [21], [30]. The correlation as a result of ETD check should be looked at with caution, as only 5 data-points are available.

Table 4.15: Pearson correlation matrix (sub) processes security checkpoint; experimental data

	Baggage drop	to WTMD	Between drop-reclaim	Baggage reclaim	Security checkpoint	Group size	Number of X-ray boxes	Experience	Occupation	2x WTMD	ETD check
Baggage drop	0	0	0	0	0.4538	0	0.4584	0	0	0	0
to WTMD	0	0	0.4082	0	0	0.4286	0.3961	-0.5100	0	0.3874	0
Between drop-reclaim	0	0.4082	0	0	0	0	0	0	0	0	0
Baggage reclaim	0	0	0	0	0.8735	0	0	0	0	0	0
Security checkpoint	0.4538	0	0	0.8735	0	0	0.3773	-0.4037	0	0	0
Group size	0	0.4286	0	0	0	0	0	-0.4427	0	0.4737	0
Number of X-ray boxes	0.4584	0.3961	0	0	0.3773	0	0	-0.4117	0	0.3929	0
Experience	0	-0.5100	0	0	-0.4037	-0.4427	-0.4117	0	0	-0.3790	0
Occupation	0	0	0	0	0	0	0	0	0	0	-0.3894
2x WTMD	0	0.3874	0	0	0	0.4737	0.3929	-0.3790	0	0	0
ETD check	0	0	0	0	0	0	0	0	-0.3894	0	0

A similar correlation matrix can be constructed for the simulation. Here, not 30 entries are used, but the 11733 simulated passenger. The resulting Pearson correlation matrix can be seen in Table 4.16. It should be observed less parameters are presented as the model does not allow for inclusion of all available processes. Here also, the security checkpoint is influenced strongly by the baggage drop (0.6855). The occupation did not have a significant impact on any process during the experiment, while is found to be of influence during simulation. The results are small ($\rho < 0.3$). Of the small influences, the largest is associated with the security checkpoint time, being (0.1837).

With the influencing of parameters known, a closer look can be taken at trying to make the model behave more like reality. This governing model output is the throughput which is 2.1060 for the simulation and 1.36 for the experiment. The largest influencing factor for throughput is expected to be security checkpoint time, which is the time between entering (from queue to baggage drop) and leaving the security checkpoint (finished baggage drop and walking away from the checkpoint as more boxes clog the system and less passengers can drop/reclaim baggage simultaneously). This is influenced - among others - through the number of x-ray boxes (as the number of boxes influences the drop time and thus security checkpoint time. Use of lesser

Table 4.16: Pearson correlation matrix (sub) processes security checkpoint; simulated data

	Baggage drop	to WTMD	Between drop-reclaim	Baggage reclaim	Security checkpoint	Occupation
Baggage drop	0	-0.3850	-0.2719	0	0.2829	0.1580
to WTMD	-0.3850	0	0.6364	0	0.2558	0.1409
Between drop-reclaim	-0.2719	0.6364	0	0	0.5643	0.1085
Baggage reclaim	0	0	0	0	0.6855	0.0476
Security checkpoint	0.2829	0.2558	0.5643	0.6855	0	0.1837
Occupation	0.1580	0.1409	0.1085	0.0476	0.1837	0

boxes makes x-ray scanning the baggage more difficult and prone to errors). This is in turn influenced by 2x WTDM. It is found that more experienced passenger, using more x-ray boxes, tend to go through the WTMD twice more often, as they take of shoes only when specifically asked. Hence, inclusion of the 2x WTMD might tune the model to represent reality better. The time required for these double WTMD passes are added to the baggage drop time. A new distribution is fitted and implemented into the model, which throughput is analysed. This route is chosen as parameters such as experience and number of boxes is difficult to quantitatively give a time-component, while the 2x WTMD provides regular - not subjected to inference - time-labelled data. Furthermore, it can be argued that experience and the number of boxes becomes visible in the timing data of subprocesses within the security checkpoint lane. Hence, these are indirectly incorporated in the model.

When the 2x WTMD is added to the baggage drop data a re-fitted Weibull distribution - however still passing the goodness-of-fit test - is greatly surpassed by the Gamma distribution (Smirnov-Kolmogorov test 0.4563 vs 0.8393). This can be seen in the next Table 4.17.

Table 4.17: Baggage drop input distribution, iteration 0 and 1

	Distribution	Mean	Standard deviation
Baggage drop + 2x WTMD Experiment	Gamma: Scale: 6.8212 Shape: 8.4931	57.9331	19.8790
Baggage drop Original Experiment	Weibull: Scale: 58.1001 Shape: 3.6867	52.4223	15.8308

The new average throughput during saturation becomes 2.0435 with a confidence interval of $[2.0282, 2.0589]$. The inclusion of the 2x WTMD does provide an incremental improvement over the original case, which was found to be $[2.0901, 2.1219]$ with a mean of 2.1060.

Another process which could be included through the baggage drop waiting time is the 'to WTMD' time. This is time standing idle, waiting before one can pass through the WTMD. As this happens before the WTMD check, it is intuitive to add this to the baggage drop time. Now, again, a Gamma distribution is preferred using the goodness-of-fit test (rank 1 of 19).

Table 4.18: Baggage drop input distribution, iteration 0 and 1 and 2

	Distribution	Mean	Standard deviation
Baggage drop + 2x WTMD Experiment + to WTMD	Gamma: Scale: 7.3324 Shape: 9.3057	68.2331	22.3677
Baggage drop + 2x WTMD Experiment	Gamma: Scale: 6.8212 Shape: 8.4931	57.9331	19.8790
Baggage drop Original Experiment	Weibull: Scale: 58.1001 Shape: 3.6867	52.4223	15.8308

The new average throughput during saturation becomes 1.9156 with a confidence interval of $[1.9032, 1.9281]$. The inclusion of the 2x WTMD does provide an incremental improvement over the original case, which was

found to be [2.0901, 2.1219] with a mean of 2.1060 and first iteration with a mean of 2.0435 with a confidence interval of [2.0282, 2.0589]. The experimental throughput is 1.36 passengers per minute.

Three remaining subprocesses which can be translated to the model are the time between the WTMD and baggage reclaim, the additional check due to the WTMD and an ETD check. Each of these processes - and time it takes to perform these - are added incrementally to the baggage reclaim time. This will be done in the same way as done with the two iterations done before and the baggage drop time. Firstly, the time between going through the WTMD to start baggage reclaim is added. Now, the distribution changes to a Gamma distribution, as it performs slightly better than the normal distribution (and in line with the distribution directive), 4.20.

Table 4.19: Baggage reclaim input distribution, iteration 0 and 1

	Distribution	**Mean**	**Standard deviation**
Baggage reclaim + from WTMD to reclaim	Gamma: Scale: 5.0486 Shape: 16.1165	81.3658	20.2678
Baggage reclaim Original Experiment	Normal: Mean: 58.1333 Standard deviation: 37.2331	58.1333	37.2331

As can be seen from Table 4.19, the mean is significantly moved to the right (longer time). The results are profound for the throughput. The throughput is significantly lowered to a mean of 1.6916 and a confidence interval of [1.6832, 1.7000].

Finally, the additional search due to failing to pass the WTMD check and ETD checks are added to the baggage drop time. This is done simultaneously, as only one WTMD check is performed and a separate iteration is not expected to yield significantly different results. Again, a Gamma distribution is found, with slightly updated parameters, as can be seen in Table 4.20.

Table 4.20: Baggage reclaim input distribution, iteration 0, 1 and 2

	Distribution	**Mean**	**Standard deviation**
Baggage reclaim + from WTMD to reclaim + WTMD check + ETD check	Gamma: Scale: 5.6891 Shape: 14.8647	84.5668	21.9342
Baggage reclaim + from WTMD to reclaim	Gamma: Scale: 5.0486 Shape: 16.1165	81.3658	20.2678
Baggage reclaim Original Experiment	Normal: Mean: 58.1333 Standard deviation: 37.2331	58.1333	37.2331

This last iteration only slightly improves the throughput model output. This is to be expected as both checks are not performed often and are of short duration. The throughput now becomes [1.6467, 1.6639] with a mean of 1.6553. Although all available data and subprocesses are added to the input distributions of the model, still a large difference can be observed in the throughput with respect to the experimental data, being 1.36 passengers per minute.

The other model output of interest is the security checkpoint time, again, is the time between entering and leaving the security checkpoint. In the next Table 4.21 the throughput and security checkpoint time during each iteration of model input modification is presented. Firstly, a graphical analysis of the data found using the last iteration is conducted. As can be seen in Figure 4.7, a positive skew in the data can be observed. Therefore, normality conditions are violated and a Normal distribution cannot be fitted, as was possible for the experimental data. For security checkpoint analysis, the best fitting distribution is used without regard for the distribution directive. In practice, this means the Generalised Extreme Value distribution is used as this can be modified by three parameters. This makes this distribution the most 'agile' and able to represent the data best. The results are summarised in Table 4.21. The data in the table suggests that the throughput can be reduced, at the cost of an increased security checkpoint time. This increased checkpoint time greatly exceeds

the experimental security checkpoint time, which was found to be 145 seconds, see Figure 4.6. It can be seen that even the baseline model overestimates the security checkpoint time. A potential cause might lay in the independence assumption of the baggage drop en baggage reclaim processes, as could already be detected for the baseline model in Table 4.14. After all, the mean time for baggage reclaim is higher than the input distribution. Hence, a closer look at the input distributions and realised distributions through the iterations is to be analysed. Although for the baseline model comparison, input and output distributions were held constant (as seen in Table 4.14), now, also the most agile distribution is chosen as for the security checkpoint time, being the generalised extreme value distribution. It is found that both processes display output characteristics very similar to the input distributions. For drop, an experimental mean of 68.9929 and a standard deviation of 29.2606 is found, while for the baggage reclaim a mean of 88.3060 and a standard deviation of 22.7868 is found. This is close to the input values found as the top entry of Table 4.18 and 4.19. The similarity between the input and realised distributions of the baggage drop and baggage reclaim, and the lack of similarities between experimental and realised distribution of the security checkpoint time can - among others - be attributed to the fact that due to walking of passengers in the simulator, time is represented in the model twice (for example during from baggage drop to WTMD and from WTMD to baggage reclaim).

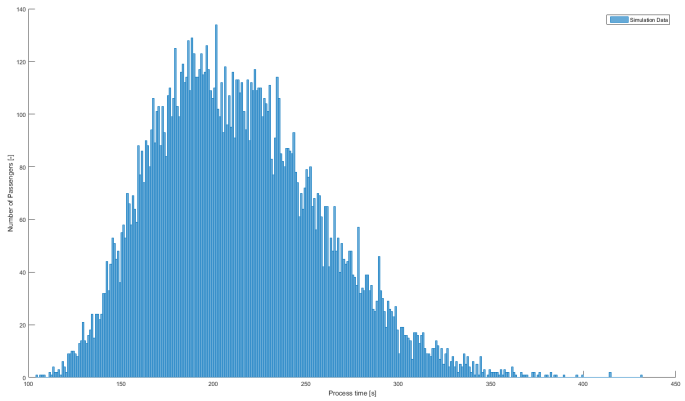


Figure 4.7: Security checkpoint duration 128 simulations using the final iterated model, iteration 4

Table 4.21: Output analysis AATOM through the iterations with 3 baggage drop and 3 baggage reclaim places

	Parameters Generalised Extreme Value	**Mean**	**Standard deviation**	**Throughput**
Baseline	shape=-0.1284 scale = 46.558 location = 144.0500	165.6241	51.8843	2.1060 [2.0901, 2.1219]
Iteration 1	shape = -0.1483 scale = 47.1742 location = 147.091	168.2166	51.6612	2.0435 [2.0282, 2.0589]
Iteration 2	shape = -0.1399 scale = 46.3725 location = 152.2850	173.3484	51.1500	1.9156 [1.9032, 1.9281]
Iteration 3	shape = -0.1041 scale = 38.1978 location = 189.5760	208.0279	43.5443	1.6916 [1.6832, 1.7000]
Iteration 4	shape = -0.1204 scale = 39.7260 location = 194.6340	213.2959	44.5939	1.6553 [1.6467, 1.6639]

Even with the inclusion of processes twice in the model and a far greater security checkpoint time in the simulator compared to reality, the throughput found using AATOM is still higher than found experimentally. Hence, a modification to the model is proposed. As mentioned before, the occupation is much higher in the model compared to the experiment. Also, the mean security checkpoint time is now much higher than in the

experiment. To get this to represent reality better, the three baggage drop and reclaim spots are reduced to 2, yielding a total maximum occupation of 4, compared to the original 6 available places. It is found that before the WTMD, the occupation was 2.60745. After the WTMD, the occupation is on average 2.5697. When these values are combined, a value greater than the average total security filter occupancy is found. Hence, both values are floored to 2 to arrive at the total experimental mean occupation. The even split between baggage drop and baggage reclaim of both having two places available is chosen as both experimental values found are rather similar, and no indication presented itself an asymmetric split should be used.

This modification - the fifth iteration - is implemented into the model. The effect of this change in the model is assessed firstly using the baseline model and its input distributions as found in Table 4.12. Now, the average throughput is 1.4013 with a confidence interval of [1.3937, 1.4090]. This already approaches the experimental value of 1.36. The occupation of the system also approaches the experimental value 3.5987, with a simulated occupation of 3.7624 and a confidence interval [3.7503,3.7745]. The security checkpoint time however is still higher than experimental (mean of 162.6123, standard deviation 52.3102). A possible explanation might be the arrival rate. It is hypothesised that the simulation overestimates the queue length, as could be observed when simulations where run with the graphical representation enabled, while in practice the arrival rate is closer to the saturation point, possibly reducing waiting times in the simulator and hence the security checkpoint time. No experimental data was gathered for assessing the queue length, hence the model output for queue length could not be used for model analysis with respect to the experimental data. The arrival 'rate' is inspected per minute during saturation, and graphically displayed in a Histogram 4.8. In AATOM, the arrival of passengers is now specified to a fraction of the total passengers per minute. Hence, in the 1st minute (30th for the experimental data), 3/30 of the total passengers are spawned. This is the sixth iteration.

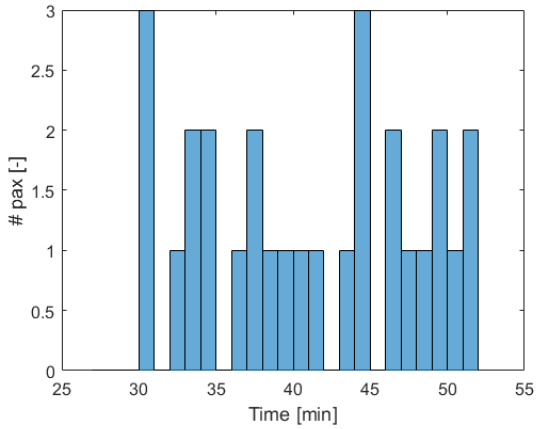


Figure 4.8: Passenger arrival per minute during saturation, between 30 and 52 minutes

It should be noted that the arrival rate replicated is the arrival rate at the security checkpoint baggage drop. Hence, it is an approximation of the actual arrival rate at the airport. Given the available data, this is the most accurate way of defining the actual arrival rate. What should also be observed is that the agents have to walk towards the security checkpoint in AATOM. This walking is of negligible influence, as it will merely translate the results forward in time, while the influence on the arrival rate is limited. Again, 128 replications with different seed are performed. In this interval, an average of 31.0313 passengers are spawned per simulation with a confidence interval of [30.3005,31.7620]. When looking at the throughput between 5 and 19 minutes (between model initialisation and ending of the shortest simulations, as can be seen in Figure 4.9), a mean of 1.3961 passengers per minute is found, with a confidence interval of [1.3742,1.4180]. The empirical throughput is slightly smaller than the lower confidence interval bound, with a value of 1.36. The system occupation is very similar as well. Experimentally, the occupation is 3.5987. The simulated occupation is [3.4411,3.5564] with a mean of 3.4988. Next, a closer look is taken at the security checkpoint times. A mean checkpoint time of 163.2933 is found with a standard deviation of 51.9627 for the generalise extreme value. This means the security checkpoint time is still approximately 20 seconds higher than found experimentally. When looking at the input distributions, the realised values are for baggage drop a mean of 43.0764 and a standard deviation of 25.1095. The baggage reclaim has a mean of 61.9131 and a standard deviation of 34.8356. As can be seen when looking at the input distributions found using the experimental data in Table 4.6, the realised distribu-

tions are faster than the experiment would suggest. Hence, another factor is yet not taken into account which is of influence on the security checkpoint time. A parameter of influence could be the walking velocity of the passengers. Up until now, the passengers walked at 1 m/s. It can be found that this on the lower bound of passenger velocity [44]. Here, an average velocity of $1.34 m/s$ is found, with a peak of 1.6 m/s for passengers aged 20. Therefore, the walking velocity is increased by 34%. The mean is used as no data is recorded about the age of the passengers and gives the alteration for the seventh iteration.

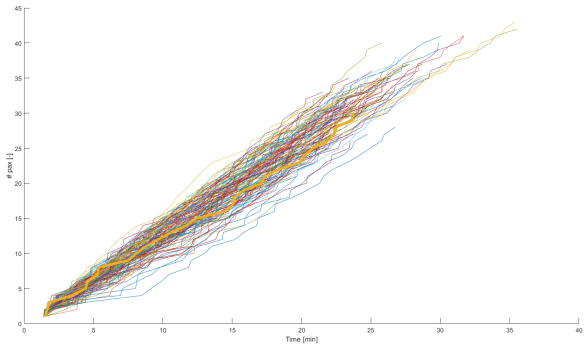


Figure 4.9: Throughput simulated passenger during saturation April 17 08:30 - 08:50 at the security checkpoint at RTHA with replicated arrival rate. Iteration 6, 128 Replications. The thick yellow line is the experimental data

With the walking velocity altered, the throughput rises slightly to 1.5703 and a confidence interval of [1.5450, 1.5955]. The occupation drops slightly, as due to faster walking, the average persons in the system per minute drops to 3.2199 and a confidence interval of [3.1507, 3.2891]. The baggage drop also drops with 5 seconds to 38.7196 and a standard deviation of 21.7079. The baggage reclaim remains approximately the same with a mean of 60.5063 and standard deviation of 32.3918. Finally, the checkpoint time is reduced significantly to 134.7196 seconds for the mean and a standard deviation of 40.5383. As can be seen, the walking velocity greatly influence the security checkpoint time at little detrimental effects on throughput and occupation. As the security checkpoint now suffers from an undershoot, additional processes can again be added to the input distributions during baggage drop and baggage reclaim. This is done through inclusion of the additional checks during baggage drop en baggage reclaim. In order to be able to re-evaluate the individual contribution to the change in model performance, this too is done iteratively, starting with the inclusion of a second WTMD check, as seen in Table 4.18, resulting in the eight iteration. The throughput now has a mean of 1.5064 and a confidence interval of [1.4843, 1.5286], which is lower than without the altered baggage drop distribution. The occupation has risen to 3.2466 with a confidence interval of [3.1820, 3.3112]. The mean and standard deviation of drop and reclaim time have risen to 42.6711, 23.9710 and 60.1839, 32.4223. The mean checkpoint time is now increased to 139.5943 with a standard deviation of 42.0652. The inclusion of the 2 WTMD check makes the model behave more like the actual dataset. Now, also the checks associated with baggage reclaim are implemented. A different distribution compared to Table 4.19 is used as the 'from WTMD to reclaim' is omitted and only the additional checks are incorporated. The result of the inclusion of the post WTMD pass-through checks can be seen in Table 4.22 which is the basis for the ninth iteration.

Table 4.22: Baggage reclaim input distribution, iteration 0 and 9

	Distribution	**Mean**	**Standard deviation**
Baggage drop + WTMD check + ETD check	Weibull: Scale: 69.1737 Shape: 1.8173	61.4876	35.0431
Baggage reclaim Original Experiment	Normal: Mean: 58.1333 Standard deviation: 37.2331	58.1333	37.2331

This ninth iteration reveals that the throughput is still too high with a mean of 1.4957 and a confidence interval of [1.4707, 1.5207]. The occupation is still on the low side with a mean of 3.3121 and a confidence interval of [3.2485, 3.3756]. Finally, the mean checkpoint time approaches the experimental data closer with a mean

time of 142.9127 and a standard deviation of 41.7182. Ventures with different walking velocities (1.25 and 1.20 m/s respectively) did not provide improvements. The throughput can also be depicted graphically in the next Figure 4.10.

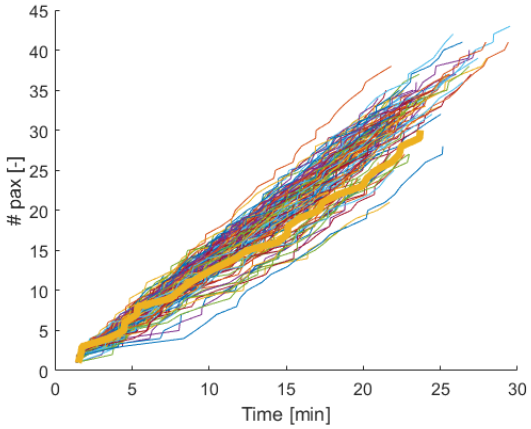


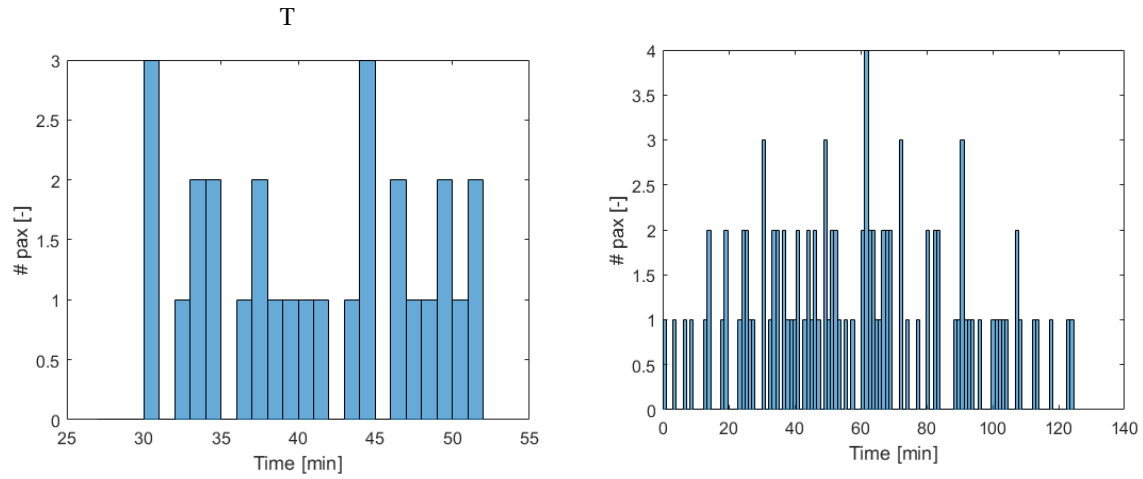
Figure 4.10: Throughput simulated passenger during saturation April 17 08:30 - 08:50 at the security checkpoint at RTHA with replicated arrival rate. Iteration 9, 128 Replications. The thick yellow line is the experimental data

As no other data available can be included to improve the distribution fitting nor a case can be made that the walking velocity can be altered due to a lack of age-data, this 9th iteration and its characteristics is used for subsequent cases. This can be summarised as follows:

1. Constrain occupation simulator to match the experimental data
2. Set the walking velocity to 1.34 m/s
3. Set experimental data as arrival input ('rate')
4. Include the time required to pass the WTMD more than once to the baggage drop time, before distribution fitting
5. Include the time required to perform a WTMD and/or ETD check to the baggage reclaim time, before distribution fitting

4.4.3. AATOM: A Simple Case, Extended

Now, a closer look is taken at the entire bank, during the morning of April 17, 2018. Given the tuning during saturation, the performance of AATOM for a two-hour period is investigated by keeping the model parameters the same apart from the arrival input. To this end, the arrival rate of the entire bank is used and fed into the model. Here, the distributions for baggage drop and baggage reclaim are held constant as found for iteration 9 and the saturated case, while only the arrival distribution is scaled-up. These results can be found in the next Figure 4.12. Hence, iteration nine (inclusion of additional checks to the baggage drop and baggage reclaim with a walking velocity of 1.34/ms) is used. The arrival in the simulations is constructed in an identical fashion as before. This can be seen in Figure 4.11, where passengers spawning is specified per minute.



(a) During saturation, 8:30 - 8:50

(b) Full bank, 8:00 - 10:02

Figure 4.11: Passenger arrival input April 17, (full) morning bank

Upon closer inspection of this Figure 4.12, it can be seen that the general shape resembles the experimental data, but overperforms as passengers are faster processed compared to the experimental data. Hence, other factors change as well with a different loading of the system, other than the arrival rate. Put differently, saturated performance is not able to describe the security filter on a broader range through only changing the arrival of passengers. An intuitive modification required to make the simulation represent reality better is to modify the input distributions for baggage drop and baggage reclaim. To this end, distributions are fitted to the available 102 passengers, which is an extension of the available 30 passengers.

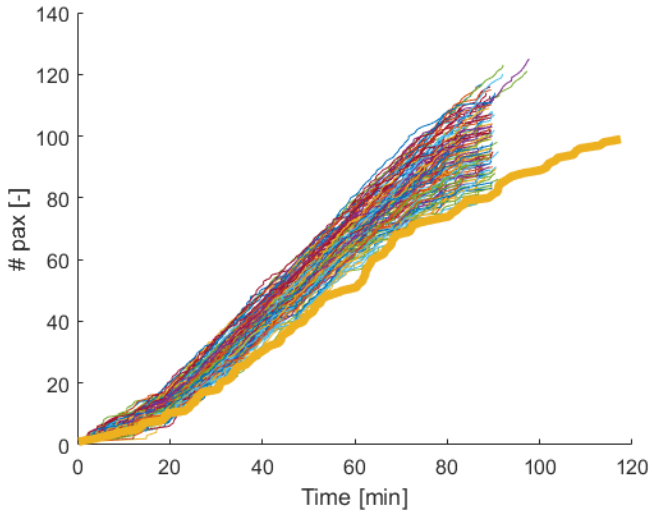
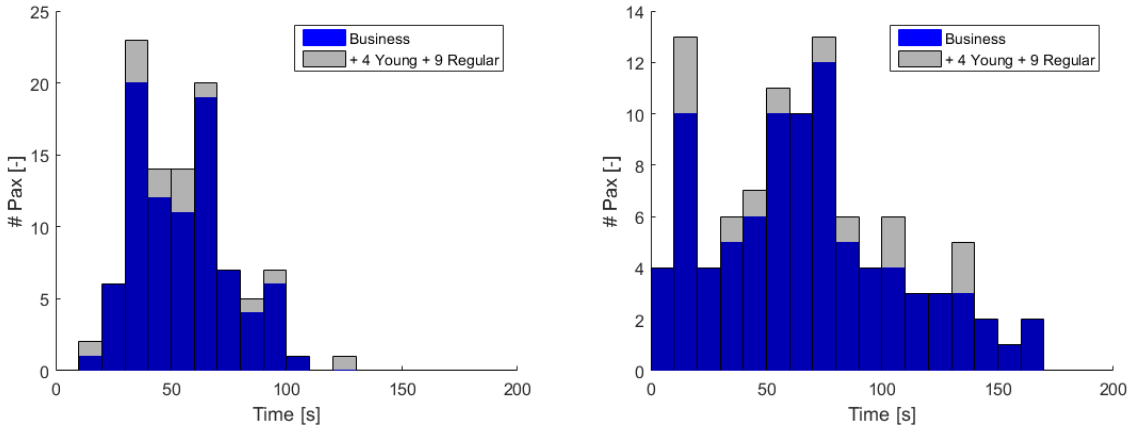


Figure 4.12: Throughput simulated passenger during saturation April 17 08:30 - 08:50 at the security checkpoint at RTHA with replicated arrival rate. Iteration 9, 128 Replications. The thick yellow line is the experimental data

Once more, a general passenger is constructed using all the available data. This passenger also is primarily business and a bit of regular and young. The histograms can be seen in Figure 4.13.



(a) Baggage drop

(b) Baggage reclaim

Figure 4.13: Histogram baggage drop/reclaim April 17, full morning bank

Added to these times are 2x WTMD, WTMD checks and the ETD checks. This results in these following distributions, as can be seen in Table 4.23. Observe the input distribution for baggage drop being a Generalised Extreme Value distribution. The distribution directive did not give satisfactory goodness-of-fit results, requiring the stray away from the preferred subset.

Table 4.23: Selected input distributions based on data gathered April 17, morning during security checkpoint saturation

	Distribution Type	**Mean**	**Standard deviation**
Baggage drop	Generalised Extreme Value shape = 0.2710 scale = 22.3574 location = 48.9786	73.4909	45.4714
Baggage reclaim	Normal: Mean: 72.8075 Standard deviation: 45.7822	72.8075	45.7822

The updated fits for the larger sample size increase both the mean and standard deviation for the processes. The graphical results can be seen in Figure 4.14. The average throughput calculation for the morning flight bank of April 17 amounts to 102 passengers in 115 minutes, yielding 0.8867 pax/min. The simulation finds a mean value of 1.0972 and a confidence interval of [1.0863, 1.1081]. The experimental occupation of the simulation is found to be 2.7868 passengers. The simulated occupation has a mean of 3.0066 with a confidence interval of [2.9798, 3.0334] (the average occupancy before and after the WTMD check are 2.0118 and 1.9846. Hence, the 2x2 places for baggage drop and reclaim remains valid). The security checkpoint time is experimentally found to be 167.5916 seconds. The simulation results in a mean checkpoint time of 175.9638 and a standard deviation of 65.6812. When comparing the results found during saturation and the full flight bank is described here, it can be seen that the model is better able te represent saturation. The full day shows a larger error between the experimental and simulated data. It is hypothesised that the full day shows less homogeneous behaviour when compared to the saturated case. As such the mathematical representation of the input distributions becomes weaker which results in a larger deviation.

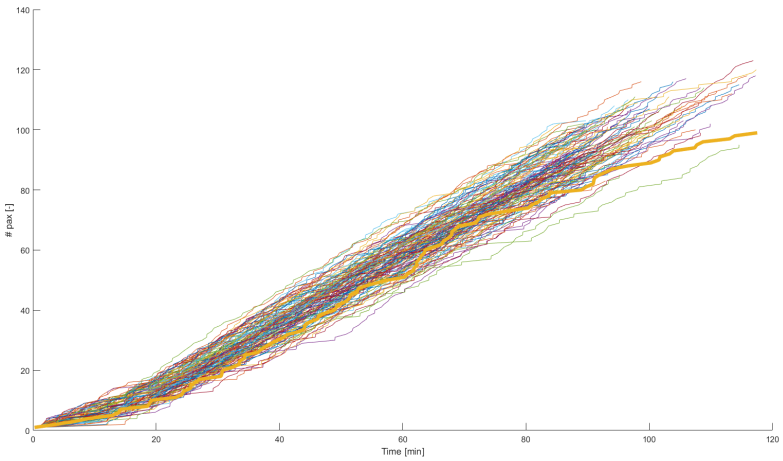


Figure 4.14: Throughput simulated passenger during the full bank April 17 08:00 - 10:02 at the security checkpoint at RTHA with replicated arrival rate. Iteration 9, 128 Replications. The thick yellow line is the experimental data

So far, the model has been investigated during saturation and scaled-up to represent a full flight bank. To this end, agents were represented using all the available data without a regard of their different structures. The following step is to try and make use of the AATOMs property to allow the creation of a heterogeneous population. This will be the investigated using a different case, case III.

4.4.4. AATOM, A Heterogeneous Case

As opposed to the 'Simple Case', now a flight bank is sought after with a diverse population. This is Case 3, as described in Table 4.3. During this day, Business, Senior, Young, and Regular passengers are recorded. As the available data is divided into smaller subsets, firstly, it should be checked whether the sample size within a passenger type remains of sufficient size to be able to draw meaningful conclusion. Hence, the statistical power of the 'goodness-of-fit' test needs to be scrutinised. To this end, an approach is used as described in [33]. Given the largest difference - $|F_(emperical) - F(experimental)|$ - between the empirical cumulative distribution function and the experimental cumulative distribution Δ_{max} - given a required significance level and a desired power as derived from Figure 4.15 - the sample size N can be derived using the following Formula 4.1.

$$N = (\frac{\text{value at 0.8 power and 95\% significance level}}{\Delta_{max}})^2 \tag{4.1}$$

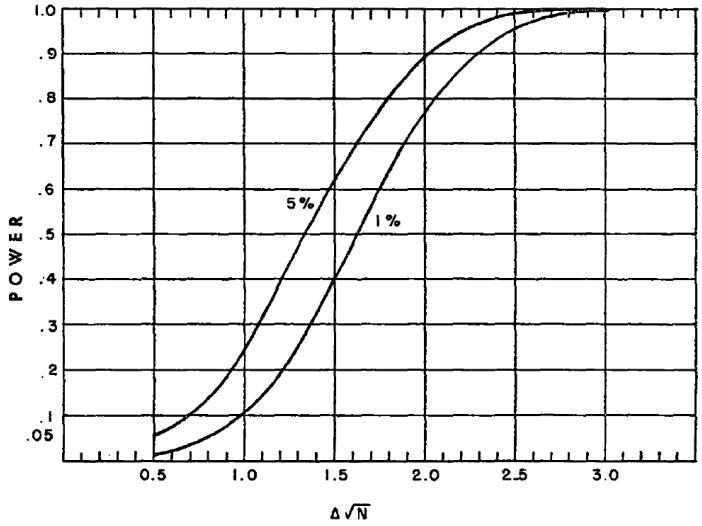


Figure 4.15: 95% and 99% significance level plot between the Kolmogorov-Smirnov Power and the sample characteristics [32]

Due to the nature of the distribution fitting - obtaining as likely as possible distribution - generally have similar cumulative distribution functions. Therefore, the maximum absolute difference between both distributions is generally (very) small. This results in a small Δ and will yield the requirement for large sample sizes when aiming at a widely applied 95% significance level and a power of 0.8 [40]. This is an intuitive observation as disproving an hypothesis becomes much harder when the data becomes more similar. As is actively tried to be as identical as possible, large sample sizes are needed to get a desired power. Therefore, no acceptable sample size - using this method - is available. To illustrate the gravity of the discrepancy, a post-hoc analysis is done on the full day of the Simple Cases which concluded Section 4.4.3. The reason a post-hoc analysis is done that the simple case was aimed at understanding the model, and to a lesser extend the validation involved.

If a closer look is taken at the distribution fits for baggage drop, baggage reclaim, and the security checkpoint time. The baggage drop fit would require 1062 samples. Baggage reclaim requires 212 samples, while the security checkpoint requires 1804 samples. As only 102 samples are available, no satisfactory power can be demonstrated.

The same problems are encountered for the identified heterogeneous case. On an individual passenger level, the sample size is in excess of a factor 5 too small to get the required power. The power problem persists when all passengers are considered simultaneously, where again a general passenger is investigated.

[32] proposes a relaxation of the conservative approach explained above in Equation 4.1. Here, an approximation is suggested with a sufficient 95% critical value - $D_{critical}$, Equation 4.2 - is satisfied when the absolute difference for the cumulative distributions functions is smaller than this critical value.

$$D_{critical} = \frac{1.36}{\sqrt{N}} \tag{4.2}$$

With this updated metric, the post-hoc analysis of the Simple Case is conducted for a second time. Now, not the sample size is considered as a starting point for assessing the power. Rather, it is tested if the critical value is larger than the largest absolute difference between the cumulative distribution functions and what sample size is considered sufficient. The critical value - since sample size based - is the same throughout and found to be 0.1347 according to Equation 4.2. The baggage drop fit passes the test as the maximum difference is 0.0545. Baggage reclaim also passes the test with a maximum difference of 0.1345. Hence, the power requirement is satisfied.

For the heterogeneous case, a similar power test is conducted to investigate if the sample sizes are of sufficient size to accept the fitted distributions. The results are listed in Table 4.24. Two distributions are fitted to the Regular passenger. This since 10 passengers are categorised belonging to two passenger groups. The passenger identified belonging to two groups are subtracted from the least specific passenger type, being 'Regular'. The distributions are selected using the distribution directive and the Smirnov-Kolmogorov 'goodness-of-fit' test, provided allowed for by the critical value.

Now, it is tried to evaluate the heterogeneous population. To this end, firstly, this general passenger is constructed and evaluated. This done through recreation of one agent, defined by the data of the recorded heterogeneous population. An identical strategy as described in Section 4.4.3 using the steps identified at the end of Section 4.4.2. The input distributions are determined again through inclusion of different processes. The recorded baggage drop time is supplemented with additional WTMD walkthrough times. The baggage drop is supplemented with additional ETD and WTMD checks. The results can be found in Table 4.24. For brevity, the histograms of the passenger types are omitted.
Firstly, the arrival input - stressor - is to be constructed. Again, using the arrival times at the security checkpoint are used. The results can be found in Figure 4.16. It can be seen that the input tensor runs from 0 to 74 minutes. Since the observations started at 05:13 - see Table 4.3 - this corresponds to the observational window of 05:13 to 06:27.

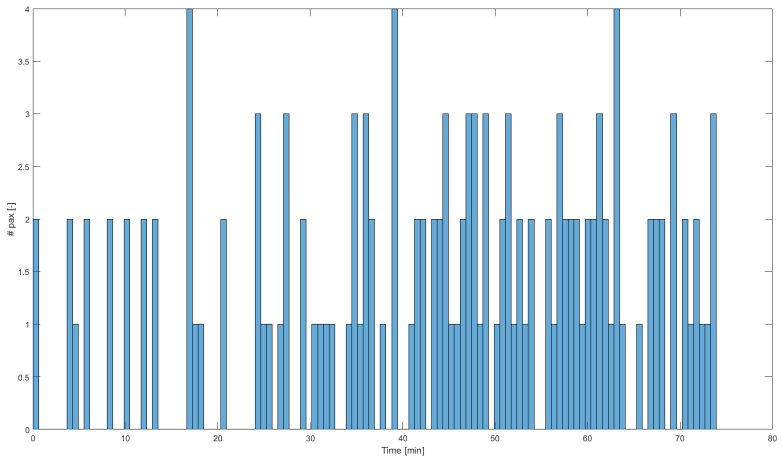


Figure 4.16: Passenger arrival input tensor March 12, full morning bank

Table 4.24: Fitted distributions per passenger type and the general passenger

Passenger Type	Process	Distribution	Mean	Standard Deviation
Business	Baggage drop	Gamma: shape: 8.9502 scale: 6.9932	62.5905	20.9215
	Baggage reclaim	Normal: mean: 65.4545 sigma: 31.8923	65.4545	31.8932
Senior	Baggage drop	Gamma: shape: 3.3800 scale: 16.9383	62.3329	32.4933
	Baggage reclaim	Gamma: shape: 2.7601 scale: 20.7719	57.3325	34.5095
Young	Baggage drop	Generalised Extreme Value: shape: -1.0284 scale: 22.8274 location: 52.8041	52.5297	23.1123
	Baggage reclaim	Generalised Extreme Value: shape: 0.4225 scale: 16.1865 location: 37.2931	58.1022	72.9648
Regular unique **(-17 business - 2 senior -2 young)**	Baggage drop	Gamma: shape: 3.8919 scale: 17.0622	66.4199	33.6680
	Baggage reclaim	Weibull: shape: 1.1035 scale: 56.4389	54.4019	49.3644
Regular combined	Baggage drop	Gamma: shape: 4.1798 scale: 15.5340	64.9290	31.7589
	Baggage reclaim	Weibull: shape: 1.8830 scale: 59.5061	52.8193	29.1516
General passenger	Baggage drop	Generalised Extreme Value: shape: -0.0641 scale: 25.3829 location: 50.9326	64.0638	30.1534
	Baggage reclaim	Generalised Extreme Value: shape: 0.1858 scale: 27.7675 location: 34.7555	56.9707	49.1327

With the stressor and input distributions known, a closer look is taken at the use of baggage drop and reclaim stations respectively. An overall occupation of 5.7282 is found, with an occupation before and after the WTMD of 2.8185 and 2.9570 respectively. Hence, for this set-up a 3x3 baggage drop and reclaim configuration in the simulator is required. Finally, a total of 160 passengers are to be spawned in the simulator, to comply with the number of passengers followed during the experiment. With these settings, 128 replications are performed. This is graphically represented in Figure 4.17.

Upon first inspection, the simulator is not able to reproduce the curve seen in the experimental data. What can be seen is that the experimental data curves from the lower end of the simulated spread to the upper end. In the first 40 minutes, the simulator over-performs with respect to the experimental data and under-performs near the end at 70 minutes. Hence, the simulated model appears to approach an average performance, whereas the experimental data suggest the performance - throughput - of the security checkpoint

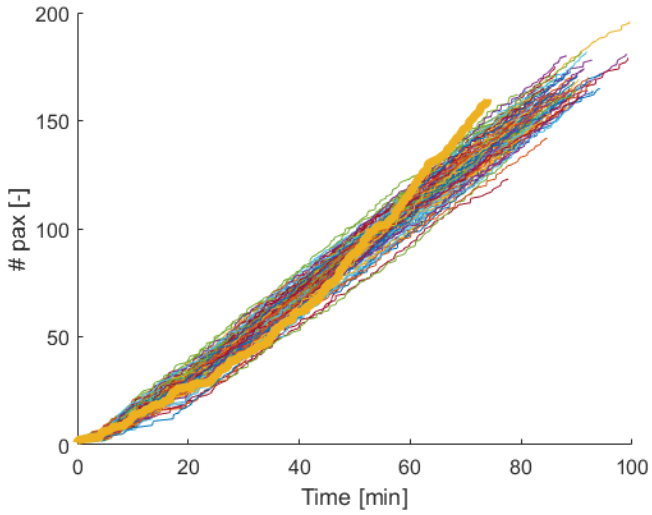


Figure 4.17: Throughput simulated passenger during the full bank March 12 05:13 - 06:27 at the security checkpoint at RTHA with replicated arrival rate tensor, using a single passenger type based on a heterogeneous experimental data

improved during the morning of March 12.

Next, the performance indicators found for the experimental data and simulated data are compared. The experimental occupation was found to be 5.7282. The simulations proved to have a mean occupation of 4.4196 and a confidence interval of $[4.3611, 4.4781]$. Next, a closer look is taken at the process rate, passengers per minute. The simulation data finds this value to be 1.9292 with a confidence interval of $[1.9002, 1.9581]$. This is slightly lower when compared to the experimental data with a value of 2.1651 passengers per minute. Finally, the security checkpoint time is considered. The simulator in this configuration results in a mean time of 150.7450 seconds and a standard deviation of 52.7704. Experimentally, this value is found to be 165.3246 seconds. What can be concluded, is a lower occupation and a lower security checkpoint time are found. A possible explanation for these values can be attributed to a shorter idle time. This idle time is the time after finishing dropping of luggage and starting baggage reclaim. The average idle time during the morning of March 12 is found to 51.0375 seconds with a standard deviation of 37.3129. From this value should the additional checks be subtracted such as multiple WTMD-passes and WTMD-checks and ETD checks. These values are on average 2.6688, 2.2062 and 1.2813 seconds. This amounts to an average value of 6.1563 seconds. Hence, the idle time not actively modelled in the simulator is $51.0375 - 6.1563 = 44.8812$ seconds on average. In the simulator, this value is found to be 25.3169 seconds with a confidence interval of $[24.8183, 25.8155]$. The difference of idle time found experimentally and in the simulator are 19.5643 seconds. This approximates the difference found in the total security checkpoint time.

It is counter intuitive to observe a lower throughput in the simulator while both occupation and security checkpoint times are lower. A possible explanation is the longer simulated time required by a significant portion of the simulation to process the passengers, and the simulation taking longer than the experimental data would suggest, amplifies the under performance of the model near the end, yielding a lower value. GUI-enabled runs of the simulator show indeed a queue is present near the end of the simulation, resulting in a longer simulation time required to dissolve these remaining passengers, while no new passengers are spawned.

Finally, a closer look needs to be taken at why the empirical data suggests the 'pitching-up'. The dataset is split at the 40 minute mark, as here the curve start to pitch up noticeably. In the first 40 minutes, the average baggage drop, baggage reclaim and security checkpoint time are 64.6287, 72.9787, and 188.7831 seconds. These are significantly higher than the average found for the complete data set. The occupation during this window is 4.4360 and the throughput is 1.5263. The Idle time is now 59.5345 seconds. From 40 minutes to the end of the observed flight bank, the times are significantly lower than the average found, yielding 63.7491, 47.7532, and 152.4401 seconds. Now, the occupation is higher with a value of 6.7724. where 4 passengers are

standing before the WTMD and 3 behind the WTMD. The throughput now becomes 2.9143. The Idle time now is reduced to 46.2059 seconds. It can be seen that the checkpoint performs significantly better after 40 minutes. When compared to the arrival input in Figure 4.16, the performance increases when the arrival of passengers increases sharply. It is hypothesised that the loading of the checkpoint directly influences its performance.

Before the extension to the heterogeneous population is implemented, it is tried to investigate the ambiguous model output of 'idle time'. Apart form being able to observe this idle time is time dependent, the data does not give clues as to where this idle time stems from. Saturation of the system seems not of major importance, as in the first 40 minutes the idle time is higher than in the remaining time window (which is markedly more busy). The comment section of the recorded data also does not contain information which might shed light on the observed (idle) timing data. Thus, it is merely observed that this idle time exists.

As such, for the extension from a general passenger to specific passenger types and a heterogeneous population, the aforementioned results for the general passenger are to be compared to the simulation results for the heterogeneous population as not all effects (time dependent distributions) can be incorporated into the model and linked to the experimental data.

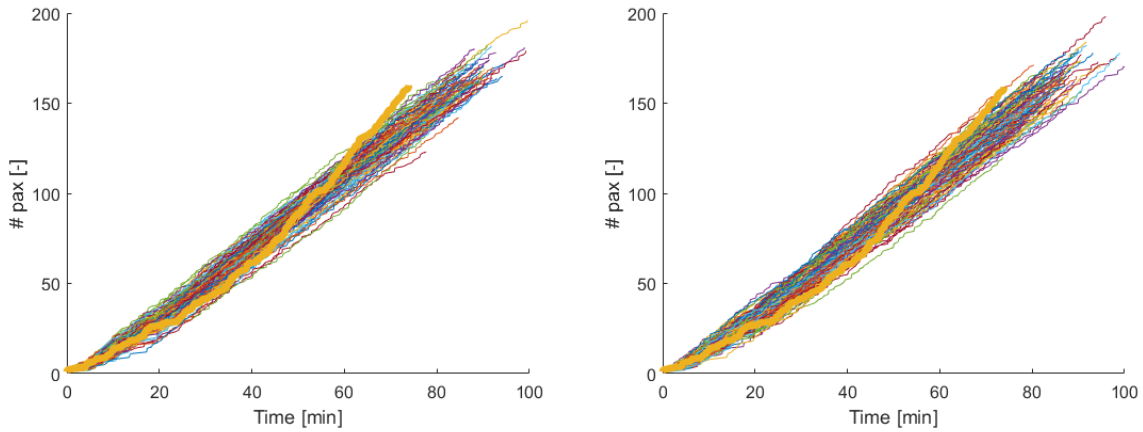
To this end, the distributions as proposed in Table 4.24 are implemented. The distribution chosen for the regular passenger is 'Passenger unique'. This as this is the most clean representation of the regular passenger, not polluted with data from other passenger types. Again, this approach is chosen as the regular category is more of a container definition, where the other passenger types are more strictly defined. As such, these passengers are considered predominantly of a more narrowly defined category. It should also be observed that two passengers types are not taken into account. Both MiVa and Family have only one data point, which is insufficient to fit a distribution. Hence, these two passengers are omitted from analysis and are also discarded in the arrival input, which is updated slightly by the removal of said passengers.

A remark should be made concerning the arrival input when the transition is made from a general passenger to specific passenger types. Although the arrival is controlled on a minute per minute basis, no coupling of passenger type per arrival instance is made. Hence, although the number of passenger per minute is controlled, the passenger type per minute is not. Therefore, only the relative occurrence of each passenger type is specified and randomly allocated to a spawned agent. The values can be seen in the Table below, 4.25.

Table 4.25: Relative occurrence of the passenger types of the full morning bank of March 12

Passenger Type	Number of Passengers	Fraction
Business	22	0.1392
Senior	27	0.1709
Young	16	0.1013
Regular	93	0.5886
AllPassengers	158	1.0000

As has become customary, firstly, a closer look is taken at the simulation results graphically. A change is made in presentation by providing a side-by-side view of the homogeneous on the left and heterogeneous case on the right, as seen in Figure 4.18. The similarities of the graphs are striking, when only graphically inspected. It does appear the variance for the heterogeneous population is larger. This is to be expected, as more variables are introduced into the model, yielding more dispersed results. Furthermore, the gradients seen similar when both configurations are compared to the yellow experimental data, which is identical for both plots.



(a) Homogeneous population

(b) Heterogeneous population

Figure 4.18: Throughput simulated passenger during the full bank March 12 05:13 - 06:27 at the security checkpoint at RTHA with replicated arrival rate, 128 replications. The thick yellow line is the experimental data.

The graphical observations are to be supported by the statistical parameters mentioned often before. The simulated throughput is 1.9185 with a confidence interval of [1.8620, 1.9749]. This is a widening of the confidence interval found for the homogeneous population, which was found to be [1.9002, 1.9581] with a mean of 1.9292. Next up is the occupation in the system. The updated simulated results yields a mean of 4.4230 and a confidence interval of [4.3425, 4.5035], which is slightly more than for the homogeneous case, as a mean of 4.4196 and a confidence interval of [4.3611, 4.4781] was found. Also, a closer look is taken at the idle time, which now has a mean of 25.7606 seconds and a range of [25.1382, 26.3831]. This too is a slight improvement, coming from a mean of 25.3169 and a range of [24.8183, 25.8155]. Finally, the security checkpoint time needs to be investigated. Now a mean of 149.9023 and a standard deviation of 52.5107 seconds is found. This is a performance reduction, coming from a mean of 150.7450 and a standard deviation of 52.7704. As differences with respect to the experimental data are discussed for the homogeneous case, now, a closer look is taken at the differences between both simulator settings: homogenous versus heterogeneous. As expected, the confidence intervals have increased all for the updated heterogeneous case. The means for the throughput and idle time have increased marginally, while the mean for the security checkpoint time has decreased marginally. One would expect the security checkpoint standard deviation to vary more than it does, given the larger spread. This however is not the case and also only increased marginally. To this end the passenger type composition is to be investigated. The first suspect that arises is if the fractions defined in Table 4.25 are reproduced as expected? The Business, Senior, Young, and Regular passengers in the 128 simulation runs are on average spawned with a mean fraction of 0.1390. 0.1708, 0.0990, 0.5912 and confidence intervals of [0.1331, 0.1448], [0.16550, 0.1766], [0.0940, 0.1040], and [0.5823, 0.6001]. This very much approaches the desired fractions as set in the simulator.

Although differences have been observed between the homogenous and heterogeneously populated simulated model, the magnitude is of such small scale, it is not deemed a noteworthy improvement. Is this finding a surprise? Hardly. When a closer look is taken at Section 4.2.1, it was concluded that intra-day categorisation of passenger types is not as important compared to macro-level differentiation. Furthermore, if a general passenger is defined using the combined data - by definition - very similar results will be obtained when the heterogeneous case uses the same passenger fractions using individual passenger type distributions.

The creation of the heterogeneous population is however expected to be useful when one is interested in different passenger fraction compositions under the assumption each passenger will display similar average performance throughout the simulator run time.

4.5. Different Passenger Fractions

In total, 2277 passengers were followed through the security checkpoint at RTHA. In addition to the discussed normal operation an experiment was conducted - of which the experimental design can be found in Appendix B - where passenger were directed to one of two open security checkpoint lanes in the security checkpoint. B. This resulted in a different fraction of passenger type per security lane. One lane was used mostly by faster passenger (such as business passengers) while the other lane was used mostly by slower passengers (such as families), categorised based on appearance. It was found that in practice this unequal distribution of passenger among different lanes in the security checkpoint yielded different security lane performance. It was also found that the 'slow/normal' security checkpoint setup has a better mean throughput performance than an standard security lane setup. This could be an interesting case - in further research - to try to replicate in AATOM using a heterogeneous population. An elaborate description of the results found during the experiment can be found in the paper which is in submission by Journal of Aerospace Technology and Management (JATM) [25].

5

Conclusion & Recommendations

Using experimental data gathered at Rotterdam The Hague Airport, an existing agent-based simulator AATOM is calibrated and validated. The time-labelled data is used to set the input parameters, being the baggage drop and baggage reclaim. The input parameters are set trough merging timing data. For baggage drop, baggage drop time and multiple walk-through metal detector times passes are combined. Also, merging of data for the baggage drop process is done. Here, the baggage drop time, patdown time, and explosive trace detection time is combined.

The unexpected important variables were found to be the number of locations where agents are able to perform the baggage drop and baggage reclaim. Although individual security checkpoint times can be tuned correctly using the input distributions, other performance parameters cannot. Both throughput and checkpoint occupation are to be tuned using these available spaces.

As the two investigated cases have shown, reliable results can only be attained when data during that particular day is used, for baggage drop and baggage reclaim distributions, as well as the number of baggage drop and baggage reclaim locations and the arrival distribution. It is found that generalisation - even when a smaller sample is scaled up to investigate a larger time window - is difficult. Emergent behaviour - although not directly related to actual events - is expected to present itself while being useful for security checkpoint alteration analysis. One can think of the effect of using different (asymmetric) baggage drop and baggage reclaim locations or different arrival distributions for assessing throughput and occupations.

The inclusion of a heterogeneous population was not found to yield better results compared to generalised passenger implementation. It is expected to become more useful when investigating the more general phenomenon and tendencies. Categorisation has been found useful when looking at the macro-scale and it is an intuitive way of changing the general performance of the population, as different passenger types use different amounts of time for baggage drop and baggage reclaim processes. As opposed to iteratively recreating an input distribution for a general passenger, the fractions of passenger types - and its resulting performance - can be changed easily.

Another observation is that the simulator is only able to recreate the average performance of a security checkpoint. Experimental data suggest that the performance of the checkpoint is not constant. Hence, a further study into time-dependent performance should be done. A possible fusion of insights might be associated with trade-offs made by the security personnel, as described in the work by M. Jesus [27]. Also, it was found that passengers can be in the system while doing no particular task. This 'idle time' is not implemented in the model and it should be investigated how this can be done intuitively. Adding these times to either the baggage drop/reclaim input distributions is considered an option.

In total, 2277 passengers were followed through the security checkpoint at RTHA. In addition to the discussed normal operation an experiment was conducted where passenger were directed to one of two open security checkpoint lanes in the security checkpoint. This resulted in a different fraction of passenger type per security lane. It was found that in practice this unequal distribution of passenger among different lanes in the

security checkpoint yielded different security lane performance. It was also found that the 'slow/normal' security checkpoint setup has a better mean throughput performance than an standard security lane setup. This could be an interesting case to try to replicate in AATOM using a homogeneous population in future research.

A

Appendix A Formal AATOM Description

The formal description given here is directly taken from the document by the model creators [24]. The parts deemed relevant are included. The original document should be consulted for a complete overview.

A.1. Language & Architecture

This chapter describes the modelling language and architecture of AATOM. The order-sorted predicate logic-based language called LEADSTO is used [7]. This language allows both discrete and continuous modelling of a system at different aggregation levels. Furthermore, one can express both qualitative and quantitative aspects of a system using LEADSTO. The LEADSTO language is outlined in Section A.1.1. The AATOM architecture specifies different modules that are responsible for the functioning of the human agents. These are explained in more detail in Section A.1.2.

A.1.1. The LEADSTO Language

Dynamics in LEADSTO are represented as evolution of states over time. A state is characterized by a set of properties that do or do not hold at a certain point in time. To specify state properties for system components, ontologies are used that are defined by a number of sorts, sorted constants, variables, functions and predicates (i.e., a signature). For every system component A, a number of ontologies can be distinguished: the ontologies $IntOnt(A)$, $InOnt(A)$, $OutOnt(A)$, and $ExtOnt(A)$ are used to express respectively internal, input, output and external state properties of the component A. For a given ontology Ont, the propositional language signature consisting of all state ground atoms based on Ont is denoted by $APROP(Ont)$. State properties are specified based on such ontology by propositions. Propositions are formed, combining ground atoms by logical operators such as conjunction, negation, disjunction, and implication. Input ontologies contain elements for describing perceptions of an agent from the external world, such as the observed function obs: $IntOnt(A) \rightarrow APROP(IntOnt(A))$. Output ontologies describe actions and communications of agents. To this end, the function *performed*: $ACTION \rightarrow APROP(OutOnt(A))$ is introduced. Then, a state S is an indication of which atomic state properties are true and which are false: S: $APROP(Ont) \rightarrow \{true, false\}$.

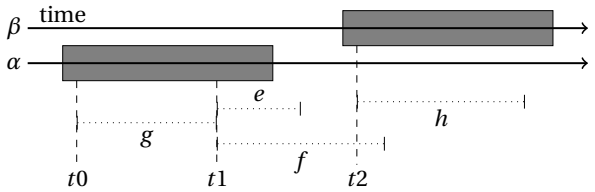


Figure A.1: Timing relationships for LEADSTO expressions.

LEADSTO enables modeling of direct temporal dependencies between two state properties in successive states, also called dynamic properties. A specification of dynamic properties in LEADSTO is executable and can be depicted graphically. The format is defined as follows. Let $\alpha 1$ and $\alpha 2$ be state properties of the form 'conjunction of atoms or negations of atoms', and e, f, g, h non-negative real numbers. In the LEADSTO language the notation $\alpha 1 \twoheadrightarrow_{e,f,g,h} \alpha 2$ means: if state property $\alpha 1$ holds for a certain time interval with duration

g, then after some delay (between e and f) state property $\alpha 2$ will hold for a certain time interval of length h (Fig. A.1). To indicate the type of a state property in a LEADSTO property we shall use prefixes $internal(c)$, $input(c)$, $output(c)$ and $external(c)$, where c is the name of a component. Consider an example dynamic property:

$$input(\mathbf{A})|obs(arrest\_fail) \rightarrow\!\!\!\!\!\rightarrow_{0,0,1,1} output(\mathbf{A})|performed(detonate())$$

Informally, this example expresses that if agent A observes a failed arrest during some time unit, then A will detonate an IED in the following time unit. Next, a *trace* or *trajectory* γ over a state ontology Ont is a time-indexed sequence of states over Ont (where the time frame is formalized by real numbers). A LEADSTO expression $\alpha 1 \rightarrow\!\!\!\!\!\rightarrow_{e,f,g,h} \alpha 2$, holds for a trace γ if:

$$\begin{aligned}&\forall t1[\forall t[t1-g \leq t < t1 \Rightarrow \alpha 1 \text{ holds in } \gamma \text{ at time } t]\\ &\Rightarrow \exists d[e \leq d \leq f \& \forall t'[t1+d \leq t' \leq t1+d+h\\ &\Rightarrow \alpha 2 \text{ holds in } \gamma \text{ at time } t']]\end{aligned}$$

More details on the semantics of the LEADSTO language can be found in [7].

A.1.2. AATOM Architecture

As human agents are central in an airport terminal, an architecture used to specify agents is specified in this work. The AATOM architecture is structured in three layers, based on the work of Blumberg [5], Hoogendoorn [22] and Reynolds [41]. In this architecture three levels of abstraction are distinguished namely, the operational layer, the tactical layer and the strategical layer. An overview of the architecture is presented in Figure A.2.

The top layer, called the *strategic layer*, is responsible for determination of goals, for updating the belief module and for reasoning. Reasoning includes analysis, planning and decision-making. The middle layer, called the *tactical layer*, is responsible for actuation of activities. It is also responsible for route navigation in the environment, the interpretation of observations and maintaining a lower level belief. The belief modules in the strategic and tactical layer exchange information with each other. The *operational layer* is responsible for interaction with the environment and other agents. Here, input is generated by sensory information an agent can observe. The output is defined as actions an agent can execute. The actions are generated based on input from the tactical layer.

Each of the different layers contain several modules that are responsible for specific tasks. In the architecture, the following main sequence of operations is followed: $observation \rightarrow perception \rightarrow interpretation \rightarrow reasoning \rightarrow activity\ control \rightarrow actuation \rightarrow action$. This reflects many of the processes discussed by Sun in his extensive analysis on cognitive architectures [47]. The different layers and modules are explained in the remainder of this chapter.

Operational Layer The operational layer is the lowest layer of the architecture and is responsible for low level interactions with the environment and other agents. It contains two modules: the perception module and the actuation module. Both modules are discussed in more detail below.

Perception Module The perception module is responsible for perceiving information from the environment or other agents. Observations of this module are further processed by the interpretation and belief module, after which other modules can access the observations. Observations for all agents are described using the following function.

$$input(\mathbf{A})|obs(\textbf{observation\_object})$$

Where $observation\_object$ represents the object that an agent observed, and obs is short for $observed$. The function maps specific $observation_object$s to a Boolean value that indicates that the specific object has been observed or not. The observations that the agents can perform are specific to agent types and are discussed in the baseline model description in Chapter A.2.

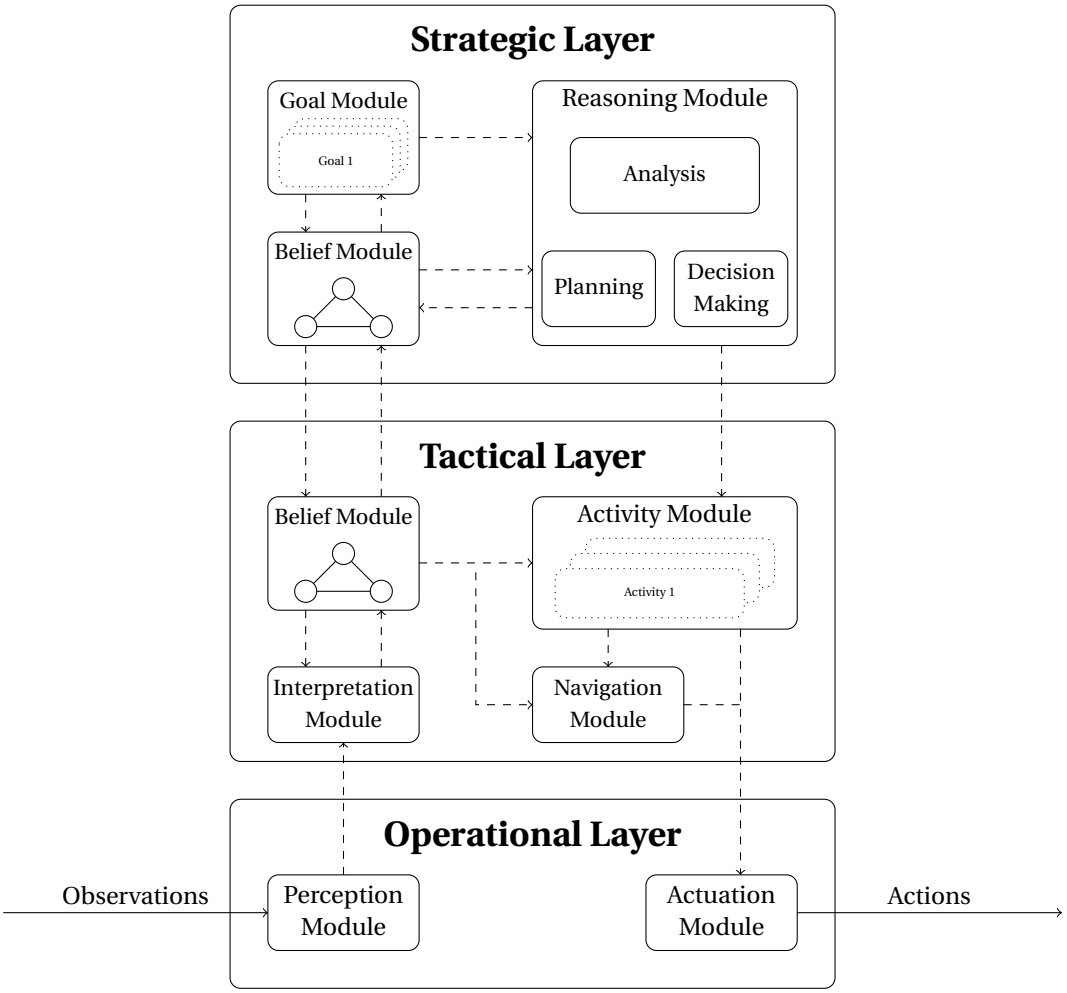


Figure A.2: The AATOM architecture.

Actuation Module The actuation module is responsible performing actions of an agent in the environment. It receives instructions from the activity module and the navigation module in the tactical layer. Within the actuation module, for instance the walking behaviour of human agents is modelled. Other actions, like communication with other agents, are also executed in the actuation module. Actions can be in any of the following three states: *not_started*, *in_progress* and *has_finished*, denoted as follows.

$$output(\mathbf{A})|action\_state(\mathbf{action\_type()})$$

Where $action\_type()$ represents the action that an agent can perform. Actions that have been performed (i.e. are in the *has_finished* state) can also be indicated using the following function.

$$output(\mathbf{A})|performed(\mathbf{action\_type()})$$

Where $performed$ states that the action has been performed. The function maps specific actions to a Boolean value that indicates that the specific action has been performed or not. The actions that the agents can perform are specific to agent types.

All human agents can communicate using a communication action. A communication is directed to another agent, has a specific type, $comm\_type$, and a related parameter, $comm\_parameter$. This is denoted as follows.

$$communicate(\mathbf{agent}, \mathbf{comm\_type}, \mathbf{comm\_parameter})$$

Tactical Layer The tactical layer is responsible for the interpretation of observations which will update the belief of the agent. Furthermore, the tactical layer is responsible for preparation of actions by executing activ-

ities. The tactical layer also determines collision-free routes to target locations. The interpretation module, activity module and navigation module are described in detail below.

Interpretation Module The interpretation module is responsible for the interpretation of observations made by the agent. It receives its input from the perception module and in cooperation with the belief module an interpretation of the observation is generated. When the interpretation is made the belief module is updated. An interpretation in this work follows the structure described below.

$$internal(\mathbf{A})|int(\mathbf{interpretation\_object})$$

Where $interpretation\_object$ is the type of interpretation that an agent has made, and int is short for $interpreted$. Often, the interpretation module is just an intermediate module that transfers information from the perception module to the belief module. In other cases, lower level observations and information in the belief module are combined into more advanced interpretations. An example of such an interpretation is the *stuck* interpretation, in which an agent determines if it is stuck in a location or not.

Belief Module The belief module is responsible for maintaining the belief of the agent. Belief is represented as a 4-tuple, outlined below.

$$\text{Belief} = \begin{pmatrix} \text{Characteristics} \\ \text{Interpretations} \\ \text{Activity \& Action State} \\ \text{Plan} \end{pmatrix}$$

Characteristics of an agent are saved in the belief module. The interpretations refer to the interpretations the agent made. When an agent interprets something different from its current belief, the belief is updated. The activity & action state is a vector of activity-state pairs and action-state pairs, containing all activities and actions the agent can execute and their corresponding states. When the status of any activity or action changes, the activity-state pair or action-state pair is updated. Finally, the plan is updated in the belief module whenever the plan is generated and/or changed in the planning module. The belief module in the tactical layer automatically exchanges its belief with the belief module in the strategic layer.

Other modules present in the architecture, like navigation and reasoning, can access the belief maintained in the belief module such that they can function properly.

Activity Module The activity module is responsible for executing activities based on input from the reasoning module in the the strategic layer and the belief module in the tactical layer. Activities are defined to be executable by agents and form a central concept in this architecture. An activity has an *activity_area* that corresponds to an area in which the activity is executed. In these areas the corresponding activities can be executed. The activity area is denoted as follows.

$$internal(\mathbf{A})|activity\_area(\mathbf{activity})$$

Similar to actions, activities can be in any of the following three states: *not_started*, *in_progress* and *has_finished*, denoted as follows.

$$internal(\mathbf{A})|activity\_state(\mathbf{activity})$$

The activity state is saved in the agent's belief module and can be accessed at any moment in time.

Navigation Module Navigation of a human agent is performed in the navigation module. The navigation module determines a collision free path towards some area. A collision free path is defined to be a path that avoids all physical objects that have the blocking property. Other passengers are not defined as physical objects and are not taken into account when calculating the path. A path is represented as a sequence of location points and saved in the $path$ variable. The function $end\_location(path)$ returns the last location point in the path.

The navigation requires input from the belief module (see Section A.3.5) to access observations in the environment and the activity module (see Section A.3.4) to determine the target area. Algorithms like the A* path finding algorithm [10] and the Jump Point Search algorithm [17] can be used for calculating the path.

Strategic Layer The strategic layer is responsible for both the determination of goals, reasoning and maintaining and updating a belief.

Belief Module The belief module in the strategic layer works the same as the belief module in the tactical layer. It exchanges information with the reasoning module and the goal module and exchanges its belief with the belief module in the tactical layer.

Goal Module The goals module is responsible for the managing of the agent's goals. A goal can be in the following states: *not_completed*, *failed* and *succeeded*, denoted as follows.

$$internal(\mathbf{A})|goal\_state(\mathbf{goal})$$

All goals start in the *not_completed* state. If at any time t all conditions of a goal are met, the goal state changes to *succeeded*. If at any time t the goal conditions cannot be met anymore, the goal states changes to *failed*. The two types of transitions of goal states from the *not_completed* state to the *succeeded* state are specified below. The transitions to the *failed* state follow a similar pattern, but occur when the *succeeded* state cannot be reached anymore.

$$internal(\mathbf{A})|goal\_state(\mathbf{goal}) = \mathbf{not\_completed} \wedge activity\_state(\textbf{\textit{some\_activity}}) = \textbf{\textit{has\_finished}}$$
$$\& \; external(\mathbf{A})|\mathbf{t} < \mathbf{t}_{goal} \twoheadrightarrow_{0,0,1,1} internal(\mathbf{A})|goal\_state(\mathbf{goal}) = \mathbf{succeeded}$$

$$internal(\mathbf{A})|goal\_state(\mathbf{goal}) = \mathbf{not\_completed} \wedge activity\_state(\textbf{\textit{some\_activity}}) = \textbf{\textit{has\_finished}}$$
$$\wedge activity\_state(\textbf{\textit{other\_activity}}) \neq \textbf{\textit{has\_finished}} \twoheadrightarrow_{0,0,1,1} internal(\mathbf{A})|goal\_state(\mathbf{goal}) = \mathbf{succeeded}$$

These goal conditions of an agent represent that the wants to finish a specific activity before a certain time or before another activity has finished. For instance, a departing passenger wants to finish its security check activity before the time of its flight.

The goals of an agent can depend on the *needs*, *values* and *motives* that are specific for an agent type.

Reasoning Module The reasoning module is responsible for the reasoning processes of an agent. It contains three modules: planning, analysis and decision making. The analysis module is responsible for the analysis of the current state of the agent. If this module determines the need for an updated plan or a decision, it communicates this to the respective modules.

Planning Module In the planning module, a plan is generated based on the goals of an agent. A plan is an ordered sequence of activities. The planning module aims to define a plan that satisfies as many of the goals of the agent as possible. A plan is generated at the moment there are no more activities left in the plan.

$$internal(\mathbf{A})|empty(\textbf{\textit{plan}}) \twoheadrightarrow_{0,0,1,1} internal(\mathbf{A})|\neg empty(\textbf{\textit{plan}})$$

Where the $empty$ function is a Boolean function that determines if a sequence contains information or not. While currently no times are related to each element of a plan are present, this can easily be extended in future versions of *AATOM*. The $next\_activity$ function returns if a specified activity is the next activity in the plan, denoted as follows.

$$internal(\mathbf{A})|next\_activity(\textbf{\textit{activity}})$$

Decision Making Module The decision making module is used to make decisions that are not covered by the other modules. For instance, while the planning module determines which activities need to be performed at what time, there can still some freedom in choosing the location. An airport can have a security area queue areas. The decision making module determines which area is chosen to execute the activity.

A.2. AATOM - Baseline Model

The AATOM baseline model is presented in this chapter. The chapter consists of three parts: environment, agents and interactions. First, the different environment objects are described in Section A.2.1. Then, the agents and their characteristics are described in Section A.3. Finally, the interactions betweeen agents and between agents and the environment are discussed in Section A.4.

A.2.1. Environment

The environment of this model is specified to be an airport terminal security checkpoint. In this chapter, the set of environment objects that form the environment is described. Three base environment objects are defined: Area, Flight and Physical Object. Some of these base components have a set of subcomponents, more specific instances of the corresponding base component. A hierarchical visualization of the environment is shown in Figure A.3. The different components of the environment are specified below.

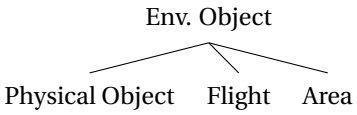


Figure A.3: The hierarchy of environment objects.

An important concept used throughout the work, is that of a *location point*. A location point is defined to be a 2 dimensional vector that contains a real valued x and y value. Location points are used throughout the model to specify locations of agents and environment objects.

Area An area is defined to be a shape that is accessible to all (human) agents. An area is a two dimensional polygon that is bounded by a sequence of location points C. Different types of areas can overlap or even completely be contained in other areas. A total of 10 different areas are defined. Each of these different areas form a direct subcomponent of the component area. A Boolean function in_area is defined to determine if a location point is in an area.

$$in\_area(\textbf{location\_point}, \textbf{area})$$

Some areas are defined to be within another area. This spatial relation between area objects can be found in Figure A.4. Each of the different types of areas that are defined are discussed below. Observe that only a closer look is taken at the entrance area, checkpoint area, and gate area, as this subset is used for this project.

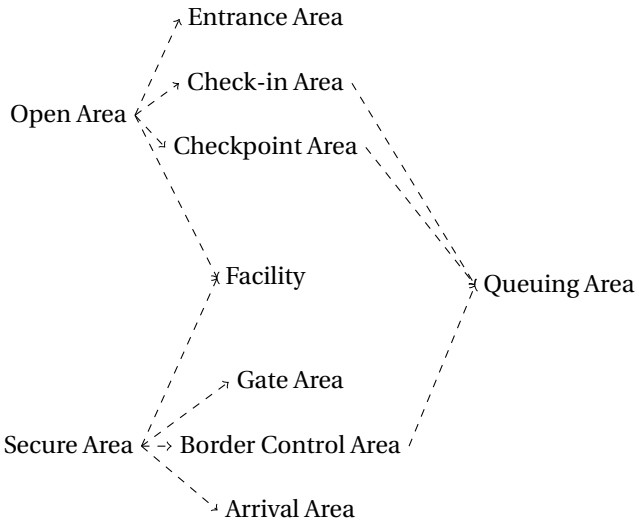


Figure A.4: The different area objects and their relative locations. An arrow pointing from one area to another implies that the second area is located within the first area. If multiple arrows point to an area, this means that that area is located in any of the areas that have an arrow pointing towards that area.

Open Area The open area is accessible for all (human) agents. This area is sometimes referred to as the non-sterile area or landside area. The open area contains physical objects and other areas. It contains at least one entrance area and a check-in area. It further contains at least one checkpoint area and can contain facility areas.

Entrance Area The entrance area is the area in which passengers enter or leave the airport terminal. In this area, passenger agents with a departure flight are generated and passenger agents with an arrival flight are removed. It is a representation of the connection between the outside world and the open area of the airport terminal.

Queue Area A queue area is a designated area for queuing. It is separated by queue separators and is used as an area in which passengers wait to be served elsewhere. It can be contained in check-in, checkpoint and border control areas.

Facility Areas There are multiple types of facility areas and they can be located throughout the airport, both in the open and in the secure area. The type of facility areas that are distinguished are shops, restaurants and bathrooms.

Checkpoint Area The checkpoint area is the area that forms the boundary between the open and secure area. It consists of a queue area and checkpoint lanes.

Checkpoint lanes contain the physical infrastructure to scan passengers. One checkpoint lane consist of an X-Ray sensor that is surrounded by belts that move luggage of passengers. Furthermore, a Walk Through Metal Detector (WTMD) is present at the checkpoint lanes. Finally, an Explosive Trace Detector (ETD) is present as well. Each of these sensors is discussed in more detail in Section A.2.1.

Secure Area The secure area (sometimes airside area or sterile area) is only accessible for human agents that have passed the checkpoint area without being stopped by any of the security operators. The secure area contains different areas: border control area, gate area and arrival gate area. These are described in more detail below. The secure area can also contain facility areas.

Gate Area A gate area is an area in which passengers will be removed from the airport terminal at the departure time of the flight (see Section A.2.1). A gate area can contain seats (see Section A.2.1).

Flight A flight is defined to be an abstract concept with the following property vector.

$$\text{Property Vector Flight} = \begin{pmatrix} \text{Flight type} & i \\ \\ \text{Time} & t \end{pmatrix}$$

- *Flight type* (i)
  - A flight is a *departure* flight.
- *Time* (t)
  - The time the flight departs.

Apart from the properties described above, a flight can also exhibit the following relations.

- $has\_gate(flight, gate\_area)$
  - The gate area that is associated with the flight. Only one gate area can be assigned to a flight.
- $is\_flown\_by(flight, passenger)$
  - A flight can be flown by a passenger. A flight can have any number of relations of this type.

Physical Object A physical object is an environment object that has a physical representation in the environment. A physical object has a sequence of location points that bounds its location. A physical object is characterized by three properties.

$$\text{Property Vector Physical Object} = \begin{pmatrix} \text{Transparency} & p \\ \text{Blocking} & b \\ \text{Sensor} & s \end{pmatrix}$$

- *Transparency* (p)
  - Boolean value that indicates if the physical object blocks the *observation* of a sensor (both the sensor of an agent and environment sensors).
- *Blocking* (b)
  - Boolean value that indicates if the physical object allows the *position* of any other blocking object or agent to be the same.
- *Sensor* (s)
  - Boolean value that indicates if the physical object is a sensor. Sensors are described in more detail in Section A.2.1.

The different types of physical objects that are defined in the model are discussed below. If a physical object property is not explicitly mentioned, it is assumed it does not possess that property.

Luggage Luggage is a transparent and non-blocking object that is described by the following property vector.

$$\text{Property Vector Luggage} = \begin{pmatrix} \text{Luggage type} & u \end{pmatrix}$$

The property vector contains three properties:

- *Luggage type* (l)
  - Luggage *carry-on* luggage.

Apart from the properties defined above, luggage is also defined by the following relations.

- $is\_owned\_by(luggage, passenger)$
  - Luggage is owned by a passenger. Luggage has exactly one relation of this type.

Queue separator Queue separators are used to form queue areas for passengers. They are located in queue areas, are blocking and transparent.

Belt A belt is present in the checkpoint area. It is used to transport luggage. A belt is blocking and transparent.

Wall Walls are used to separate the airport terminal from the outside world, but also to separate other areas from each other. Walls are blocking and non-transparent.

Sensor A sensor possesses the functionality that enables agents to sense using a mechanic object. A sensor is able to sense specific elements of the environment. An observation is conceptualized as some representation of the environment. It can be of any form- a real valued number, a Boolean value or a more complex structure. It is denoted as follows.

$$observation\_of(\textbf{sensor})$$

An observation by a sensor causes a state change of the sensor, observable by agents. This means that a sensor can be in two states: *observed* and *idle*, denoted as follows.

$$sensor\_state(\textbf{sensor})$$

A sensor can only do an observation in its idle state. When the sensor is in its observed state, after some time r_{sensor} or when an agent accessed the observation, the sensor resets its state to idle. This is formalized as follows.

$$sensor\_state(\textbf{sensor}) = \textbf{observed} \twoheadrightarrow_{0,r_{sensor},1,1} sensor\_state(\textbf{sensor}) = \textbf{idle}$$

$$input(A)|obs(\textbf{sensor}) \,\&\, sensor\_state(\textbf{sensor}) = \textbf{observed}$$
$$\twoheadrightarrow_{0,0,1,1} sensor\_state(\textbf{sensor}) = \textbf{idle}$$

Three types of sensors are defined, WTMD sensor, X-Ray sensor and ETD sensor. Each of the sensors is non-blocking and transparent.

WTMD sensor The Walk Through Metal Detector (WTMD) is a sensor that detects metal. The WTMD also randomly does an observation with a probability of p_{wtmd}. Observations of the WTMD sensor are represented as either 0, 1 or 2, where 0 is an observation of nothing, 1 is a metal observation and 2 is a random observation. A random observation can be used to start an ETD check.

X-Ray sensor The X-Ray sensor is used to observe properties of luggage. Specifically, the X-Ray sensor observes the threat level of the luggage it is observing. It is represented as a real value between 0 and 1. The threat level is compared to a threshold value to evaluate whether the threat level of the luggage is too high, and a luggage check is required. The X-Ray sensor has a processing time t_{X-Ray}. The X-Ray sensor is operated by a security employee that is assigned to perform the activity X-Ray handling.

ETD sensor The Explosive Trace Detector (ETD) is a sensor that is used to detect the presence of explosive traces. The observation of the ETD sensor is represented by a Boolean value. It can be operated by security employees with the capability to perform the activity ETD checking. The ETD has a processing time t_{etd}.

A.3. Agents

This section describes the agent types present in *AATOM* and how they operate. They follow the architecture that is presented in Section A.1.2. In Section A.3.1 the agent types are outlined, and in Section A.3.2 the human agents characteristics are described. Finally, Sections A.3.3 - A.3.5 specify how the different modules of the AATOM architecture are used by the different agents.

A.3.1. Agent Types

AATOM consists of three agent types. Namely, passengers, operators and orchestration agents. Figure A.5 presents the structure of the agent types in the environment.

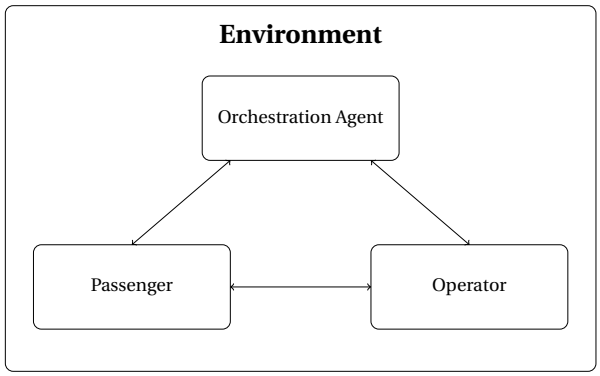


Figure A.5: The different types of agents and their interactions in AATOM.

The passenger and operator agent types represent the users and operators of the airport terminal. Additional to the passenger and operator agents, orchestration agents can be included to monitor overall performance of the airport terminal and to coordinate terminal operations. An orchestration agent can be implemented in the model and can be defined in different forms. The agent can be represented as a human agent following the three layer architecture, or as a non-human agent. Orchestration agents can be designed as such that it is able to steer operator agents and/or passengers, and make adjustments in the environment, to achieve specific system goals.

The remainder of this document describes the operator and passenger agent types in detail, and will not specify details of the orchestration agent. The definition of this agent is left general on purpose, allowing more specific versions for subsequent studies.

A.3.2. Agents Characteristics

In this section the characteristics of the human agent types, *passenger* and *operator*, are described. The specification of agent characteristics is presented by the agent's *attribute vector*, containing the agent's inherent properties, and by dynamic *relations* that hold for the specific agent type.

Human Agent All human agents have a set of shared characteristics. They have a location point, denoted as follows.

- $location\_of(agent)$
  - This relation specifies the location point of the human agent. A human agent has exactly one location point.

A human agent is represented as a circle with radius r. Other human agents and other blocking physical objects cannot be within this radius r from the location of the agent. The radius of the human agent is specified in the attribute vector below.

$$\text{Attribute Vector Human Agent} = \begin{pmatrix} \text{Radius} & r \end{pmatrix}$$

The passengers and operators defined below all have the characteristics of the human agents as well. The specific characteristics of these agents are defined in the subsequent sections.

Passenger Passengers that fly with a departure flight are referred to as *departing passengers*. To describe departing, the following attribute vectors are defined.

$$\text{Attribute Vector Departing Passenger} = \begin{pmatrix} \text{Arrival Time} & a \\ \\ \text{Checked-in} & c \end{pmatrix}$$

The properties are described as follows:

- *Arrival time* (a)
  - The arrival time for a passenger is the time it arrives at the airport.

Apart from the attributes defined above, the following relations for passengers are specified.

- $flies\_with(passenger, flight)$
  - This relation specifies the flight that the passenger has. A passenger can only fly with exactly one flight. This is the inverse relation of $is\_flown\_by(flight, passenger)$.
- $owns(passenger, luggage)$
  - This relation specifies the luggage that the passenger owns. A passenger can own any number of luggage. This is the inverse relationship of $is\_owned\_by(luggage, passenger)$.

Security Operator The security operator is of the agent type *operator*. To describe the security operator, the following relation defined.

- $assigned\_to(operator, activity)$
  - The assignment of a security operator is defined as the activity the operator is currently performing. The activities the operator can execute are specified in Section A.3.4.

A.3.3. Operational Layer

The operational layer.

Perception Module
Observations can be performed by all agents and are specified by the observation function (see Section A.1.2). The different types of observations that can be made by passengers are outlined below.

- $obs($**current_area**$)$
  - Passengers can observe in what area they currently are.
- $obs($**physical_object**$)$
  - Passengers can observe physical objects in some radius r and angle ϕ.
- $obs($**human_agent**$)$
  - Passengers can observe human agents in some radius r and angle ϕ.
- $obs($**flight**$)$
  - A passenger can observe the flight f it flies with. The flight can be further used to observe relations like gate area and check-in desks.
- $obs($**luggage**$)$
  - A passenger can observe the luggage it owns.
- $obs($**wait_request**$)$

  - Passengers can be requested to wait by other agents.

Security operators can do several types of observations, however not all observations are available for each of the security operators present within the model. Based on the assignment of the specific security operator, different types of observations are available.

- obs(**passenger**)
  - The security operators performing the luggage drop activity, the physical check activity, the ETD check activity, and the travel document check activity can observe a passenger at the corresponding activity area.
- obs(**luggage**)
  - This observation can be performed by a security operator performing the luggage check activity.
- obs(**x_ray_sensor**)
  - This observation can be performed by the security operator responsible for the X-Ray activity. The security operator observes an X-Ray sensor and can observe properties like the sensor state or observation it last performed.
- obs(**wtmd_sensor**)
  - This observation can be performed by the security operator responsible for the physical check activity. The security operator operator observes an WTMD sensor and can observe properties like the sensor state or observation it last performed.
- obs(**etd_sensor**)
  - This observation can be performed by the security operator responsible for the ETD check activity. The security operator observes an ETD sensor and can observe properties like the sensor state or observation it last performed.
- obs(**search_request**)
  - The search communication requests the operator to search the luggage that is closest.

Actuation Module The actuation module is responsible performing the actions in the environment. Within the actuation module, the walking behaviour of human agents is modelled. Walking behaviour is used to transition between activities and to complete activities. The walking mechanism of a passenger is based on the Social Force Model developed by Helbing et al. [19]. The model combines physical and 'social' forces to calculate the velocity and velocity changes of a passenger. The velocity changes depend on observations of human agents and physical objects, within the passenger's proximity. The module requires input from the navigation moduleand the interpretation & perception module. This walking action is denoted as follows.

*move(**location_point**)*

Passengers can execute the following actions.

- $wait$($\mathbf{t}_{wait}$)
  - Passengers can perform the wait act for t_{wait} seconds. This ensures that a passenger does not move for the specified time.
- $drop\_luggage$(**luggage**)
  - Passengers can drop luggage, either at a desk or belt.
- $collect\_luggage$(**luggage**)
  - Passengers can collect luggage at a belt.

Security operators can execute the following actions.

- $search$(**luggage**)
  - The search action can be performed by a security operator responsible for the luggage check activity.
- $check\_physical$(**passenger**)
  - The check physical action can be performed by a security operator responsible for the physical check activity.
- $check\_etd$(**passenger**)
  - The check ETD action can be performed by a security operator responsible for the ETD check activity.
- $check\_x\_ray$(**x_ray_sensor**)
  - The check X-Ray action can be performed by a security operator responsible for the X-Ray activity.

Finally, check-in operators can execute the following actions.

- $check\_in$(**passenger**)
  - The search action can be performed by a security operator responsible for the luggage check activity.

A.3.4. Tactical Layer

Interpretation Module The interpretation module in AATOM is mostly used for the simple transfer of observations of the perception module to the belief module. Furthermore, all agents can determine if they are stuck, based on current and historic observations. They do this by updating a stuck timer in the belief module based on the time spent at a certain location. This leads to the following state transition.

$$internal(\mathbf{A})|t_{stuck} > t_{threshold} \twoheadrightarrow_{0,0,1,1} internal(\mathbf{A})|int(\textbf{stuck})$$

Passengers can also observe if an activity area is occupied or not. They do this by counting the number of passengers that they observed and comparing it to the number of available places.

$$\begin{aligned} &internal(\mathbf{A})|next\_activity(\textbf{activity}) \wedge int(activity\_area(\textbf{activity})) \wedge int(\mathbf{B}_1) \wedge \ldots \wedge int(\mathbf{B}_n) \\ &\& external(\mathbf{A})|in\_area(location\_of(\mathbf{B}_1), activity\_area(\textbf{activity})) \wedge \ldots \\ &\wedge in\_area(location\_of(\mathbf{B}_n), activity\_area(\textbf{activity})) \twoheadrightarrow_{0,0,1,1} internal(\mathbf{A})|int(\textbf{activity\_area\_full}) \end{aligned}$$

Where B_i is some observed passenger, and n is the maximum number of passengers that can execute an activity in the area.

Activity Module The activity module is responsible for preparing the activities an agent will execute based on input from the reasoning module in the the strategic layer. The specification of the functioning of the activity module is specified in Section A.1.2. How an agent chooses between the different activities is discussed in Section A.3.5. Here, the different activities of both passengers and operators are discussed in detail.

Passengers Departing passengers can execute a total of two activities. Each of these activities are outlined below.

Queue Activity The queue activity is executed when other activities cannot be executed due to capacity limitations at the activity area. It can be performed by all passengers that want to start another activity and the queue activity is not bound to a specific area, but is often performed in a queue area. It is modelled as a (set of) waiting period(s) that end(s) when the passenger in front moves forward, or when the planned next activity has an available space again. Formally, the starting condition of the queue activity is defined as follows.

$$internal(\mathbf{A})|int(\textbf{activity\_area\_full}) \twoheadrightarrow_{0,0,1,1} internal(\mathbf{A})|activity\_state(\textbf{\textit{queue\_activity}}) = \textbf{\textit{in\_progress}}$$

Once the activity started, the following is executed.

$$internal(\mathbf{A})|activity\_state(\textbf{\textit{queue\_activity}}) = \textbf{\textit{in\_progress}} \twoheadrightarrow_{0,t_{wait},1,1} output(\mathbf{A})|performed(wait(\mathbf{t}_{wait}))$$

$$internal(\mathbf{A})|activity\_state(\textbf{\textit{queue\_activity}}) = \textbf{\textit{in\_progress}} \wedge \neg int(\textbf{activity\_area\_full}) \twoheadrightarrow_{0,0,1,1} internal(\mathbf{A})|activity\_state(\textbf{\textit{queue\_activity}}) = \textbf{\textit{has\_finished}}$$

Checkpoint Activity The checkpoint activity is mandatory for all departing passengers. It is executed at a checkpoint area. The checkpoint activity consists of several actions, of which two mandatory.

1. $drop\_luggage(\textbf{luggage})$
2. $collect\_luggage(\textbf{luggage})$

Both of these actions can be preceded by an optional waiting activity, if the action cannot be executed at the moment. Furthermore, passengers perform a wait action, when communicated by security operators.

1. $wait(\mathbf{t_{wait}})$

The checkpoint activity starts when the following conditions holds.

$$internal(\mathbf{A})|int(activity\_area(\textbf{\textit{checkpoint\_activity}})) \wedge \neg int(\textbf{\textit{activity\_area\_full}}) \wedge next\_activity(\textbf{\textit{checkpoint\_activity}}) \twoheadrightarrow_{0,0,1,1} internal(\mathbf{A})|activity\_state(\textbf{\textit{checkpoint\_activity}}) = \textbf{\textit{in\_progress}}$$

After the activity started, the luggage drop action is executed if the following conditions hold.

$$internal(\mathbf{A})|activity\_state(\textbf{\textit{checkpoint\_activity}}) = \textbf{\textit{in\_progress}} \wedge int(\textbf{\textit{luggage}}) \wedge \neg action\_state(drop\_luggage(\textbf{\textit{luggage}})) = \textbf{\textit{in\_progress}} \twoheadrightarrow_{0,t_{drop},1,1} output(\mathbf{A})|performed(drop\_luggage(\textbf{\textit{luggage}}))$$

When the drop luggage action is done, the passenger moves to the WTMD if it is not occupied.

$$output(\mathbf{A})|performed(drop\_luggage(\textbf{\textit{luggage}})) \;\&\; internal(\mathbf{A})|int(\textbf{wtmd}) \wedge location\_of(\textbf{\textit{A}}) \neq location\_of(\textbf{wtmd}) \twoheadrightarrow_{0,t_{move},1,1} output(\mathbf{A})|performed(move(location\_of(\textbf{\textit{wtmd}})))$$

When that location is reached, the passenger moves to the luggage collect area.

$$output(\mathbf{A})|performed(move(location\_of(\textbf{\textit{belt}})))$$
$$\&\, internal(\mathbf{A})|int(\mathbf{belt}) \wedge \neg location\_of(\textbf{\textit{A}}) = location\_of(\mathbf{belt})$$
$$\twoheadrightarrow_{0,t_{move},1,1} output(\mathbf{A})|performed(move(location\_of(\textbf{\textit{belt}})))$$

Once the location of the belt has been reached, the passenger performs the luggage collect action.

$$output(\mathbf{A})|performed(drop\_luggage(\textbf{\textit{luggage}}))$$
$$\&\, internal(\mathbf{A})|int(\mathbf{belt}) \wedge \neg location\_of(\textbf{\textit{A}}) = location\_of(\mathbf{belt})$$
$$\wedge \neg action\_state(collect\_luggage(\textbf{\textit{luggage}})) = \textbf{\textit{in\_progress}}$$
$$\twoheadrightarrow_{0,t_{collect},1,1} output(\mathbf{A})|performed(collect\_luggage(\textbf{\textit{luggage}}))$$

If the agent receives a communication by another agent to wait, it will do that.

$$internal(\mathbf{A})|int(\textbf{\textit{wait\_request}})$$
$$\twoheadrightarrow_{0,t_{wait},1,1} output(\mathbf{A})|performed(wait(\mathbf{t}_{wait}))$$

The activity is finished, when the luggage has been collected and the agent is not waiting.

$$internal(\mathbf{A})|activity\_state(\textbf{\textit{checkpoint\_activity}}) = \textbf{\textit{in\_progress}}$$
$$\&\, output(\mathbf{A})|action\_state(wait(\mathbf{t}_{wait})) \neq \mathbf{in\_progress}$$
$$\twoheadrightarrow_{0,0,1,1} internal(\mathbf{A})|activity\_state(\textbf{\textit{checkpoint\_activity}}) = \textbf{\textit{has\_finished}}$$

Security Operator As defined in Section A.3.2, a security operator has an assignment. The assignment defines which activity the security operator is currently assigned to. Furthermore, a security operator can do observations. All activities and observations that can be executed by security operators are defined below.

When the activity is in progress, and the operator did not communicate with the passenger yet, the operator communicates a wait order to the passenger.

$$internal(\mathbf{A})|activity\_state(\textbf{\textit{tdc\_activity}}) = \textbf{\textit{in\_progress}}$$
$$\&\, output(\mathbf{A})|\neg performed(communicate(\mathbf{passenger}, \mathbf{wait}, \mathbf{t}_{wait}))$$
$$\twoheadrightarrow_{0,0,1,1} output(\mathbf{A})|performed(communicate(\mathbf{passenger}, \mathbf{wait}, \mathbf{t}_{wait}))$$

After the passenger waited, the activity is over.

$$internal(\mathbf{A})|activity\_state(\textbf{\textit{tdc\_activity}}) = \textbf{\textit{in\_progress}}$$
$$\&\, output(\mathbf{A})|performed(communicate(\mathbf{passenger}, \mathbf{wait}, \mathbf{t}_{wait}))$$
$$\twoheadrightarrow_{0,t_{wait},1,1} internal(\mathbf{A})|activity\_state(\textbf{\textit{tdc\_activity}}) = \textbf{\textit{has\_finished}}$$

Luggage Drop Activity The luggage drop activity is currently modelled as a holder activity. As passengers currently drop their luggage on their own (within the checkpoint activity), no action is needed from the security operator performing the luggage drop activity. In future versions of AATOM, the luggage drop activity can be modelled as an an interaction with passengers that do not execute their $drop\_luggage$ action properly.

X-Ray Activity Luggage is checked by going through the X-Ray sensor. This process is executed by an security operator and defined as the X-ray activity. The X-Ray activity starts when the security operator observes that the X-Ray sensor is in the observed state. The operator observes the X-Ray sensor state and the observed the output of the X-Ray sensor, which is a real number representing the threat level of the luggage.

$$internal(\mathbf{A})|int(\textbf{\textit{x\_ray\_sensor}}) \wedge sensor\_state(\textbf{\textit{x\_ray\_sensor}}) = \mathbf{observed}$$
$$\twoheadrightarrow_{0,0,1,1} internal(\mathbf{A})|activity\_state(\textbf{\textit{checkpoint\_activity}}) = \textbf{\textit{in\_progress}}$$

If the output of the X-Ray sensor exceeds a threshold h, the X-Ray activity includes an interaction with the operator for the luggage checking.

$$internal(\mathbf{A})|activity\_state(\textbf{\textit{checkpoint\_activity}}) = \textbf{\textit{in\_progress}} \wedge int(\textbf{\textit{x\_ray\_sensor}}) \wedge observation\_of(\textbf{\textit{x\_ray\_sensor}}) > \mathbf{h} \rightarrow_{0,0,1,1} internal(\mathbf{A})|activity\_state(\textbf{\textit{checkpoint\_activity}}) = \textbf{\textit{has\_finished}}$$

If the observed X-Ray sensor observation is below the threshold h, the activity is completed without any further action. The luggage will proceed to the luggage collect area.

$$internal(\mathbf{A})|activity\_state(\textbf{\textit{checkpoint\_activity}}) = \textbf{\textit{in\_progress}} \wedge int(\textbf{\textit{x\_ray\_sensor}}) \wedge observation\_of(\textbf{\textit{x\_ray\_sensor}}) < \mathbf{h} \wedge int(\mathbf{operator}) \wedge int(\mathbf{luggage}) \rightarrow_{0,0,1,1} internal(\mathbf{A})|activity\_state(\textbf{\textit{checkpoint\_activity}}) = \textbf{\textit{has\_finished}} \,\&\, output(\mathbf{A})|performed(communicate(\mathbf{operator}, \mathbf{search}, \mathbf{luggage}))$$

Luggage Check Activity If the security operator performing the luggage check activity observes the search communication, the luggage check activity is started. This activity is modelled as an interaction between the security operator responsible for the luggage check and the passenger.

$$internal(\mathbf{A})|activity\_state(\textbf{\textit{luggage\_check\_activity}}) \neq \mathbf{in\_progress} \wedge int(\textbf{\textit{search\_request}}) \wedge int(\textbf{\textit{luggage}}) \rightarrow_{0,0,1,1} internal(\mathbf{A})|activity\_state(\textbf{\textit{luggage\_check\_activity}}) = \textbf{\textit{in\_progress}}$$

$$internal(\mathbf{A})|activity\_state(\textbf{\textit{luggage\_check\_activity}}) = \textbf{\textit{in\_progress}} \wedge int(\mathbf{luggage}) \wedge int(owner\_of(\mathbf{luggage})) \rightarrow_{0,0,1,1} output(\mathbf{A})|performed(communicate(owner\_of(\mathbf{luggage}), \mathbf{wait}, \mathbf{t}_{wait}))$$

$$internal(\mathbf{A})|activity\_state(\textbf{\textit{luggage\_check\_activity}}) = \textbf{\textit{in\_progress}} \,\&\, output(\mathbf{A})|performed(communicate(owner\_of(\mathbf{luggage}), \mathbf{wait}, \mathbf{t}_{wait})) \rightarrow_{0,t_{wait},1,1} internal(\mathbf{A})|activity\_state(\textbf{\textit{luggage\_check\_activity}}) = \textbf{\textit{has\_finished}}$$

Physical Check Activity The security operator performing the physical checking activity is responsible for detecting illegal items on the body of a passenger. It performs a check based on observations of the WTMD, the check is initiated when the WTMD sensor output value is 1.

$$internal(\mathbf{A})|int(\textbf{\textit{wtmd}}) \wedge sensor\_state(\textbf{\textit{wtmd}}) = \mathbf{observed} \rightarrow_{0,0,1,1} internal(\mathbf{A})|activity\_state(\textbf{\textit{physical\_check\_activity}}) = \textbf{\textit{in\_progress}}$$

$$internal(\mathbf{A})|activity\_state(\textbf{\textit{physical\_check\_activity}}) = \textbf{\textit{in\_progress}} \wedge int(\textbf{\textit{wtmd}}) \wedge sensor\_state(\textbf{\textit{wtmd}}) = \mathbf{observed} \wedge observation\_of(\textbf{\textit{wtmd}}) = \mathbf{1} \rightarrow_{0,0,1,1} output(\mathbf{A})|performed(communicate(\mathbf{passenger}, \mathbf{wait}, \mathbf{t}_{wait}))$$

$$internal(\mathbf{A})|activity\_state(\textbf{\textit{physical\_check\_activity}}) = \textbf{\textit{in\_progress}} \,\&\, output(\mathbf{A})|performed(communicate(\mathbf{passenger}, \mathbf{wait}, \mathbf{t}_{wait})) \rightarrow_{0,t_{wait},1,1} internal(\mathbf{A})|activity\_state(\textbf{\textit{physical\_check\_activity}}) = \textbf{\textit{has\_finished}}$$

ETD Check Activity The security operator performing the ETD checking activity is responsible for detecting explosive traces on a passenger. The check is based on indications of the WTMD, the check is initiated when the WTMD sensor output value is 2. The process is modelled as an interaction with a passenger and a waiting period of the passenger.

$$internal(\mathbf{A})|int(\textbf{\textit{wtmd}}) \wedge sensor\_state(\textbf{\textit{wtmd}}) = \textbf{observed}$$
$$\twoheadrightarrow_{0,0,1,1} internal(\mathbf{A})|activity\_state(\textbf{\textit{etd\_check\_activity}}) = \textbf{\textit{in\_progress}}$$

$$internal(\mathbf{A})|activity\_state(\textbf{\textit{etd\_check\_activity}}) = \textbf{\textit{in\_progress}} \wedge int(\textbf{\textit{wtmd}})$$
$$\wedge sensor\_state(\textbf{\textit{wtmd}}) = \textbf{observed} \wedge observation\_of(\textbf{\textit{wtmd}}) = \mathbf{2}$$
$$\twoheadrightarrow_{0,0,1,1} output(\mathbf{A})|performed(communicate(\textbf{passenger}, \textbf{wait}, \mathbf{t}_{wait}))$$

$$internal(\mathbf{A})|activity\_state(\textbf{\textit{etd\_check\_activity}}) = \textbf{\textit{in\_progress}}$$
$$\& output(\mathbf{A})|performed(communicate(\textbf{passenger}, \textbf{wait}, \mathbf{t}_{wait}))$$
$$\twoheadrightarrow_{0,t_{wait},1,1} internal(\mathbf{A})|activity\_state(\textbf{\textit{etd\_check\_activity}}) = \textbf{\textit{has\_finished}}$$

A.3.5. Strategic Layer

Here, the strategic layer for departing is described. As the behaviour of operator agents follows directly from the individual activity they perform, the plans and goals of these agents consists only of a single item (the activity they perform). Therefore, a strategic layer for these agents is not defined.

Goals Module The goals module.

Departing Passenger A departing passenger has at most five goals that it wishes to achieve. These goals are related to the activities it can execute, outlined below.

$$\text{Activities} = \begin{pmatrix}\text{Security activity}\\ \text{Gate activity}\end{pmatrix}$$

There are two goals that are the same for every departing passenger. The conditions for these goals are stated below.

$$internal(\mathbf{A})|activity\_state(\textbf{\textit{gate\_activity}}) = \textbf{\textit{has\_finished}} \,\&\, external(\mathbf{A})|\mathbf{t} < \mathbf{t}_{flight}$$

$$internal(\mathbf{A})|activity\_state(\textbf{\textit{checkpoint\_activity}}) = \textbf{\textit{has\_finished}}$$
$$\wedge activity\_state(\textbf{\textit{gate\_activity}}) \neq \textbf{\textit{has\_finished}}$$

The first goals states that the passenger wants to finish its gate activity before the time of its flight, while the second goal states that the security activity has to be finished before the gate activity.

Reasoning Module The reasoning module is responsible for the reasoning processes of an agent. It contains three modules: planning, analysis and decision making. The analysis module is responsible for the analysis of the current state of the agent. If this module determines the need for an updated plan or a decision, it communicates this to the respective modules. In this model, the analysis module is not used and only serves as a placeholder. The other two models are discussed below.

Planning Module When a departing passenger arrives in the airport terminal (i.e. at $t = t_{arrival}$, it will generate a plan containing a complete set of activities it has to perform to fulfill its goals. A plan is defined by the set of selected activities and the order of the activities. The activities are selected from the following set.

Departing Passenger When a departing passenger arrives in the airport terminal, it will generate a plan containing a complete set of activities it has to perform to fulfill its goals. A plan is defined by the set of selected activities and the order of the activities. The activities are selected from the following set.

The activity order, for each passenger, is based on the goals that are defined in the goals module. All activities have a fixed position in the sequence of activities, apart from the facility activity. To determine the position of this activity in the plan, an integer value n is drawn randomly from the set $n \in N$ with $N = [1, M]$. Here, the value M represents the total number of activities present in the plan, not taking into account the facility activity. Then, the value of n represents on which position in the plan the facility activity is located. For instance, a passenger needs to go through security and to the gate, then $M = 2$. During the generation of the plan a random number is drawn from the set $N = [1,2]$. If this number is 2, the facility activity will be executed as second activity, thus after the check-in activity is completed. If this position is infeasible, the position is removed from the set N, and the process is repeated.

Belief Module The belief module is responsible for maintaining the belief of the agent. The belief is represented as a triple, outlined below.

$$\text{Belief} = \begin{pmatrix} \text{Interpretations} \\ \text{Activity State} \\ \text{Plan} \end{pmatrix}$$

The interpretations refer to the interpretations the agent made. When an agent observes something different from its current belief, the belief is updated. The activity state is a vector of activity-state pairs, containing all activities the agent can execute and their corresponding states . When the status of any activity changes, the activity-state pair is updated. Finally, the plan is updated in the belief module whenever the plan is generated and/or changed in the planning module.

Other modules present in the model, like navigation and reasoning, can access the belief maintained in the belief module such that they can function properly.

A.4. Interactions

Additional to the autonomous behaviour described in Chapter A.3, agents interact with each other and the environment.

A.4.1. Environment

This section describes the interactions defined between agents and the environment defined for this model.

Security Operator interaction with Sensors As was described in Section A.2.1, sensors can be in two states: *observed* and *idle*. When a security operator accesses the observation, the sensor resets its state to idle. This interaction occurs between the three sensor types (WTMD sensor, X-Ray sensor and ETD sensor) and their operators.

$$input(\mathbf{A})|obs(\textbf{\textit{sensor}}) \,\&\, external(\mathbf{A})|sensor\_state(\textbf{\textit{sensor}}) = \textbf{observed} \wedge observation\_of(\textbf{\textit{sensor}}) \neq \textbf{null}$$
$$\rightarrow_{0,0,1,1} external(\mathbf{A})|sensor\_state(\textbf{\textit{sensor}}) = \textbf{idle} \wedge observation\_of(\textbf{\textit{sensor}}) = \textbf{null}$$

A.4.2. Agents

This section describes the interactions defined between agents for this model. Two types of agent interactions are defined. Interactions between operators and passengers, and interactions between security operators. These are discussed in the following sections.

Operator - Passenger Interaction All interactions between passengers and operators follow the same structure: the passenger is instructed to wait for a specified time period by the operator (performing a specific activity) under a certain condition.

The main structure of an interaction between an operator and a passenger in this work is outlined below.

$$internal(\textbf{operator})|activity\_state(\textbf{current\_activity}) = \textbf{in\_progress} \wedge int(\textbf{passenger})$$
$$\rightarrow\!\!\!\!\rightarrow_{t_{wait},t_{wait},1,1} output(\textbf{passenger})|performed(wait(\mathbf{t}_{wait}))$$

In the outline above, when some operator is performing activity and observes a passenger in its activity area, it communicates a waiting instruction to the observed passenger. This waiting instruction consists of a waiting time t_{wait}, which is followed by the passenger.

The following interactions between operators and passengers are defined. The activity that operator is performing is indicated between brackets.

- Security Operator (Luggage Drop) with Passenger
- Security Operator (Luggage Check) with Passenger
- Security Operator (Physical Check) with Passenger
- Security Operator (ETD Check) with Passenger

Operator - Operator Interaction One interaction between operators is defined in the model. The interaction is between the security operator responsible for X-Ray scanning and the security operator responsible for luggage checking. When the security operator responsible for X-Ray scanning observes a threat level above h, it communicates a search command to the security operator responsible for luggage checking. This operator then starts its luggage check activity.

$$internal(\mathbf{A})|activity\_state(\textbf{x\_ray\_activity}) = \textbf{in\_progress} \wedge int(\textbf{x\_ray})$$
$$external(\mathbf{A})|sensor\_state(\textbf{\textit{x\_ray}}) = \textbf{observed} \wedge observation\_of(\textbf{\textit{x\_ray}}) > \mathbf{h}$$
$$\rightarrow_{0,0,1,1} output(\mathbf{B})|performed(search(\textbf{luggage}))$$

A.4.3. Coordination

As described in Section A.3.1, a global control agent can be implemented for coordination of terminal operations. In order to allow for coordination, the global control agent can be designed to interact with operators, passengers and the environment. Furthermore, coordination can be implemented by allowing specific low level interactions between passengers and operators. Then coordination can occur in a cooperative setting or in a competitive setting. In a cooperative setting, agents try to combine their efforts to accomplish as a group what they could not accomplish as individual agents. In a competitive setting, agents try to get what only some of them can have; competitive agents try to maximize their own benefit at the expense of other agents.

A.5. Input Parameters

The input parameters of the model are described in this chapter. Two types of input parameters are distinguished: environment parameters and agent parameters. These parameters are used to design the system for a specific instance.

A.5.1. Environment Parameters

Three types of environment parameters are distinguished: map layout parameters, airport parameters and sensor parameters. Each of these parameter types are discussed in more detail below.

A.5.2. Map Layout

Two map layout parameters are distinguished and listed below.

- Area locations
- Physical object locations

The area locations refers to the layout of all areas that are present without the model. The physical object locations refer to all the physical objects that are defined and placed in the model.

A.5.3. Airport Parameters

A single airport parameter is identified and showed below.

- Flight set
  - Type i
  - Time t

The flight set is the set of flights that are present within the model. For each flight in the set, the type, and time needs to be specified.

A.5.4. Sensor Parameters

Three sensor parameters are identified and listed below.

- WTMD sensor
  - Recovery time r_{wtmd}
  - Random observation probability p_{wtmd}
  - Processing time t_{wtmd}
- X-Ray sensor
  - Recovery time r_{X-Ray}
  - Processing time t_{X-Ray}
- ETD sensor
  - Recovery time r_{etd}
  - Processing time t_{etd}

As there are three types of sensors defined in the model, three types of parameters need to be determined. For each of the sensors, a recovery time r_{sensor} (the time the sensor takes to return to its idle state) and the processing time t_{sensor} is identified. For the WTMD an additional parameter, referring to the probability of random observations of the WTMD.

A.5.5. Passenger Parameters

Four passenger parameters are identified and are listed below, which define the passenger characteristics.

- Arrival time a
- Flight f
- Observation radius r
- Observation angle θ

For each passenger that is generated, seven parameters need to be determined. The first parameter, flight refers to a flight from the flight set identified in the airport parameters above. The second parameter, arrival time, refers to the time the agent arrives at the airport, and has to be before the flight time for a departing passenger and after the flight time for an arriving passenger. Finally, two observations parameters need to be set: observation radius and angle. These refer to the radius and angle the passenger can observe in.

A.6. Assumptions

This section contains a list of assumptions in this model. It contains the assumptions for each of the elements in the model. While this list is aimed to be exhaustive, it might not be complete.

- Flights have a fixed time that does not change over time.
- Sensors are 100% accurate at all times.
- Sensors are always working.
- Passengers knows in what area it is at all times.
- Passengers always travel alone.
- Passengers always follow the shortest path towards their goals.
- Security operators never deny passengers entrance to the secure area.
- All passengers pass through the WTMD.
- The goal of an agent is represented as finishing activities before a certain time.
- Passengers plans all their activities on arrival.

B

Appendix B Experimental Design

As mentioned in Chapter 4, an intervention is executed at Rotterdam The Hague Airport's Security Checkpoint in an attempt to improve performance. In this Appendix, the intervention is described and and how the experiment is be done at RTHA.

B.1. Proposed Intervention

Given the spatial, financial, and procedural constraints at RTHA - which applies to many other airports as well - a different approach to increase performance should be sought [12]. While gathering data for the current operations at the SC, it appeared passengers taking (significantly) more time to perform the tasks required at the SC than average clogged the system. Passengers were standing idle to let the slow passenger in front complete the required processes within the SC. This standing idle behind this slow passenger increases the average processing time within the SC and decreases performance. The intervention is aimed at mitigating the negative effects of these slow passengers.

The SC is composed of multiple Security Lanes. At RTHA, a maximum of three SL's can be operational simultaneously available to passengers. For the operational cases in which at least two SL's are open, an intervention can be done. This intervention tries to identify and guide the slow passengers to a slow SL. This should result in one lane with a higher percentage of slow passengers compared to the other(s). Hence, the regular SL's will suffer less from clogging, while the clogging effect of the slow lane will be less. This decreased clogging is expected as passengers which take roughly an equal amount of time to complete the required processes within the SC are more grouped together in the different lanes. Therefore - in both lanes - the idle time of passengers is reduced and throughput (efficiency) increased. But how is a slow passenger and throughput defined?

B.2. Definitions

It is essential to define what a slow passenger is. Based on data-analysis of current operations at RTHA, a performance decrease in the SL is found when a passenger takes 1.5x longer than average to put their belongings on the conveyor belt in x-ray boxes – baggage drop – in preparation of x-ray analysis. This is found through subtracting column entry 5 from column entry 3 in 'Recorded Data Per Passenger', which yields the time in seconds. It is found that on average – considering all passenger types – it takes 62 seconds do drop ones belongings in the x-ray boxes, used for x-ray analysis. When a passenger takes in excess of 90 seconds to perform the luggage drop, a faster passenger behind this slow passenger has to wait as the SL is obstructed by this slower passenger. It is this waiting that is responsible for the measurable decrease in the SL – and hence SC – throughput.

Here, throughput is defined as the amount of passengers a SC can process in a given timespan. This is equal to dividing the time frame by the average time it takes to process a passenger from the start of baggage drop to finalising baggage reclaim. If a slow passenger not only increases this time for itself, but also for the passenger(s) which are behind him/her, this average time is disproportionately increased and the throughput decreased. It is this effect that is the subject of this experiment.

Now, the definition of the slow passenger is known. Furthermore, the governing metric to measure the effect of the intervention is defined. This leads to the next question to find a way of identifying these slow passengers. After all, the passengers are to be sent to either the slow or normal lane before the actual luggage drop - the metric for defining a slow/normal passenger - is performed. This qualification has to be done based on both appearance and behaviour.

B.3. Identification Slow Passengers

As defined, a slow passenger is described as taking more than 90 seconds to drop their belongings in the x-ray boxes for x-ray analysis. During the intervention, the passengers are actively distributed between the different SL's. Here, it is tried to separate the slow passengers from the rest. In ideal circumstances all slow passengers are to be sent to the slow lane, while the rest of the passengers are sent to all other lanes except the slow lane. As this passenger allocation is done before actual baggage drop-off, inferences are to be made whether a passenger is likely to be slow or not. It is vital to formalise rules based on appearance and behaviour, to perform this inference in a methodically sound and consistent way.

To this end, a set of techniques were employed to find such indicators based on data acquired of the current operations. Both machine learning and manual evaluation of the available data has resulted in the definition of the following rules with a higher than average likelihood of being a slow passenger. Application of machine learning algorithms has identified three indicators which suggest a passenger has a twice as high likelihood of being a slow passenger, 17% without vs 35% with these indicators:

1. Family + amount of luggage to be put in 2 x-ray boxes or more
2. Senior + appearing inexperienced
3. Handicapped

As can be seen in point 2, the qualification of 'unexperienced' is added. Using the expert opinion of RTHA, inexperience leads to more interaction with the security professionals. More questions are asked and one is inclined to be less at ease. This is to be expected as a lack of experience leas to heightened stress as a passenger encounters a situation not well known to him or her.

B.4. Intervention: Active Passenger Allocation

As mentioned in 'Identification Slow Passenger', passengers with a higher probability than average of being slow are sent to slow/service SL. In ideal operations during intervention, one lane is permanently filled with correctly identified slow passengers, while the remaining lanes are fed with only remaining passengers. Furthermore, the queue lengths of all open SL's are equal in length, as to ensure the assigned passenger split remains sustainable. This is an utopian situation which is not possible in practice.

Two issues present itself. Firstly, the identification of slow passengers is subject to errors. Even when categorising the passenger exactly in accordance with the previous definitions, the inherent probability of not being a slow passengers remains great than 0%. Not exactly following the definitions, as well as passengers on the fringes of the definitions, might impact the results negatively. It is expected that the difficulties associated with classification of passengers remains relatively constant throughout the intervention.

The second problem presents itself after the passengers are as best as possible identified based on the definitions, ready to be sent to one of the open SL's. The remaining ideal situation after selection would be an arrival rate of passengers at the SC such that the ratio of remaining and slow passengers is such that the slow- and normal SL's are fed equally throughout the experiment. This equal feeding entails each lane has an arrival rate such that the resulting queue lengths are of similar length, where the slow passengers identified are exclusively sent to the slow/service lane and the remaining passenger exclusively sent to the remaining normal SL's. This is the first of four operational scenarios. Although most ideal, the chance this situation can be consistently be maintained is slim.

The second scenario is where no passengers enter the SC. The risk of such a scenario occurring can be eliminated by starting the execution of an experiment two hours before a flight-bank is scheduled as to ensure passengers are guaranteed to arrive in the chosen experimental window.

This leaves scenario three and four which are of inverse nature. The third scenario is when the arrival ratio of slow/normal passengers is such that a queue is forming for the slow SL, while the normal SL's are idle or have a substantial smaller queue.

The fourth scenario occurs when te opposite happens. Now, the slow/service SL is underutilised compared to the normal SL's. Although inverse, the third and fourth scenario suffer from the same problem which is an unequal utilisation of the available SL capacity. As the intervention at RTHA will be done during normal operations, the main goal remains to security clear passengers. It is not acceptable that the intervention increases the chance for a passenger to miss its flight.

When either situation three or four presents itself, compromises during the experiment have to made to ensure the core task of the SC is fulfilled, while still being able to perform the intervention.

- In the third scenario, slow passengers are also sent to the normal SL's, resulting in an expected decrease in performance from such lanes.
- In the fourth scenario, normal passengers are also sent to the slow/service SL, resulting in an expected increase in performance for this lane.

While suboptimal, it is expected it is still possible to get a higher than average percentage of slow passengers in the slow/service lane compared to the other available lanes, ensuring the experiment is still useful.

B.5. Current Operation & Intervention Operation

The only difference between the current and intervention operation is done by trying to separate the slow passengers from the remaining passengers. Although this appears to be making a change in only one condition, this is not the case. Multiple factors are changed to both identify a slow passengers, as well as the allocation to the slow/service lane. It should also be observed that when slow passenger separation occurs, the remaining population now differs from the current operation allocation also. Hence, no direct comparison of individual SL's can be done. A different approach of comparing two systems is chosen instead, where it is tried to find the governing performance of both systems under different operating conditions.

Despite the limitations of the setup, still, after careful data analysis it is expected inferences can be made if the operations stand to gain from the intervention. To this end, performance indicators need to be identified to assess the performance of individual SL's which combined yield the resulting system performance. Hence, it is vital to identify and define these SL performance indicators unambiguously.

Firstly, the SL performance indicators are defined. Again, these are used for individual lane performance assessment. Two categories of data types can be distinguished. Firstly, the parameters defined by ratio data are discussed, followed by the frequency defined parameter.
Security lane ratio data:

1. baggage-Drop (time between entry 3 and 5 from 'Recorded Data')
2. baggage-Reclaim (time between entry 12 and 13 from 'Recorded Data')
3. Duration additional checks: 1st and 2nd WTMD check, ETD search, and baggage search (entry 8,9 and 14,15 from 'Recorded Data')
4. Time to WTMD (time between entry 13 and 14 from 'Recorded Data')
5. Time between drop-reclaim (time between item 5 and 12 from Recorded data list)
6. Checkpoint lane time (time between item 3 and 13 from Recorded data list)
7. Number of processed passenger per time per lane (is a derivate of the aforementioned points).
8. Groupsize/Boxes

Security lane frequency data:

1. Occurrence of additional checks: 1st and 2nd WTMD check, ETD search, and baggage search (entry 8,9 and 14,15 from 'Recorded Data')

These eight performance indicators are collected for the individual lanes. Based on the SC configuration, the system performance can be derived through combining the results from the different types of SL's. The resulting performance can be investigated using these eight system (averaged) performance indicators.

Bibliography

[1] Osman Balci. Validation, Verification, and Testing Techniques Throughout the Life Cylce of a Simulation Study. *Preceedings of the 1994 Winter Simulation Conference*, 1994.

[2] Osman Balci. Validation, verification, and testing techniques throughout the life cycle of a simulation study. *Annals of operations research*, 53(1):121–173, 1994.

[3] Osman Balci, Richard E Nance, E Joseph Derrick, Ernest H Page, and John L Bishop. Model generation issues in a simulation support environment. In *Proceedings of the 22nd conference on Winter simulation*, pages 257–263. IEEE Press, 1990.

[4] Elias Bartholomew. *Airport and Aviation Security*. CRC Press, 2010. ISBN 978-1-4200-7029-3.

[5] Bruce M Blumberg and Tinsley A Galyean. Multi-level direction of autonomous creatures for real-time virtual environments. In *Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*, pages 47–54. ACM, 1995.

[6] Boeing. Current Market Outlook 2015-2034. *Boeing.Com*, page 30, 2015. URL `http://www.boeing.com/resources/boeingdotcom/commercial/about-our-market/assets/downloads/Boeing{_}Current{_}Market{_}Outlook{_}2015.pdf`.

[7] Tibor Bosse, Catholijn M Jonker, Lourens Van Der Meij, and Jan Treur. A language and environment for analysis of dynamics by simulation. *International Journal on Artificial Intelligence Tools*, 16(03):435–464, 2007.

[8] Brad M Boyerinas. Determining the statistical power of the kolmogorov-smirnov and anderson-darling goodness-of-fit tests via monte carlo simulation. Technical report, Center of Naval Analyses Arlington United States, 2016.

[9] John m. Charnes. Statistical Analysis Of Output Processes. In *1993 Winter Simulation Conference*, 1993.

[10] Xiao Cui and Hao Shi. A*-based pathfinding in modern computer games. *International Journal of Computer Science and Network Security*, 11(1):125–130, 2011.

[11] Paul K. Davis. *Generalizing Concepts and Methods of Verification, Validation, and Accreditation (VV&A) for Military Simulations,*. RAND, 1992. doi: ISBN:0-8330-1298-3. URL `http://handle.dtic.mil/100.2/ADA336851`.

[12] The Flying Dutchboy. New Central Security at Schiphol Airport, June 2015. URL `https://insideflyer.nl/schiphol-central-security/`. Accessed on 31.08.2018.

[13] F Fagiolo, P Windrum, and A Moneta. Empirical validation of agent-based models: a critical survey. 2006.

[14] Marc C Gelhausen, Peter Berster, and Dieter Wilken. Do airport capacity constraints have a serious impact on the future development of air traffic? *Journal of Air Transport Management*, 28:3–13, 2013.

[15] Jennifer Gentry, Kent Duffy, and William J Swedish. Airport capacity profiles. pages 1–E1, 2014. URL `http://www.faa.gov/airports/planning{_}capacity/profiles/media/Airport-Capacity-Profiles-2014.pdf`.

[16] David J Hand. Data mining. *Encyclopedia of Environmetrics*, 2, 2006.

[17] Daniel Damir Harabor and Alban Grastien. Online graph pruning for pathfinding on grid maps. In *Twenty-Fifth AAAI Conference on Artificial Intelligence*, 2011.

[18] Douglas M Hawkins. The problem of overfitting. *Journal of chemical information and computer sciences*, 44(1):1–12, 2004.

[19] Dirk Helbing and Peter Molnar. Social force model for pedestrian dynamics. *Physical review E*, 51(5): 4282, 1995.

[20] M Hellmann. Fuzzy logic introduction" a laboratoire antennes radar telecom. *FRE CNRS*, 2272, 2001.

[21] Andy Hole and Graham Field. *How to Design and Report Experiments*. SAGE Publications, London, 2003. ISBN 9789004310087. doi: 10.15713/ins.mmj.3.

[22] Serge P Hoogendoorn and Piet HL Bovy. Pedestrian route-choice and activity scheduling theory and models. *Transportation Research Part B: Methodological*, 38(2):169–190, 2004.

[23] IATA. Checkpoint of the Future. (2013.08.07):1–102, 2012. URL http://www.iata.org/whatwedo/security/pages/checkpoint-future.aspx.

[24] Stef Janssen, Anne-Nynke Blok, and Arthur Knol. Kolmogorov smirnov test & power of tests. Department of Statistics, University of Oxfordl, . 29 May 2019.

[25] Stef Janssen, Régis van der Sommen, Alexander Dilweg, and Alexei Sharpanskykh. Data-driven analysis of airport security checkpoint operations. In submission JATMl, .

[26] Stef Janssen, Anne-Nynke Blok, and Arthul Know. Aatom - an agent-based airport terminal operations moodel, September 2016.

[27] Marson S. Jesus. A methodological underpinned modelling approach to identify, analyse and evaluate trade-offs based on empirical choice observations. Master's thesis, Delft University of Technology.

[28] Franziska Klügl. A validation methodology for agent-based simulations. *Proceedings of the 2008 ACM symposium on Applied computing - SAC '08*, page 39, 2008. doi: 10.1145/1363686.1363696. URL http://portal.acm.org/citation.cfm?doid=1363686.1363696.

[29] A.M. Law and W.D. Kelton. *Simulation modeling and analysis*. McGraw-Hill series in industrial engineering and management science. McGraw-Hill, 2000. ISBN 9780070592926. URL https://books.google.nl/books?id=QqkZAQAAIAAJ.

[30] Averill M. Law and Dativ W. Kelton. *Simulation Modeling and Analysis*. McGraw Hill, 1991. ISBN 0070592926. doi: 10.1145/1667072.1667074.

[31] C. M. MacAl and M. J. North. Tutorial on agent-based modelling and simulation. *Journal of Simulation*, 4(3):151–162, 2010. ISSN 17477778. doi: 10.1057/jos.2010.3. URL http://dx.doi.org/10.1057/jos.2010.3.

[32] S. Massa. Kolmogorov smirnov test & power of tests. Department of Statistics, University of Oxfordl. Revision 2 February 2016.

[33] Frank J Massey Jr. The kolmogorov-smirnov test for goodness of fit. *Journal of the American statistical Association*, 46(253):68–78, 1951.

[34] MatchWorks. fitdist. URL https://www.mathworks.com/help/stats/fitdist.html. Accessed on 31.01.2019.

[35] Adolf D. May and Hartmut E.M. Keller. A deterministic queueing model. *Transportation Research*, 1(2): 117–128, 1967. ISSN 00411647. doi: 10.1016/0041-1647(67)90167-0.

[36] S A Mumayiz. Overview of airport terminal simulation models. *Transportation Research Record*, 1273: 11–20, 1990.

[37] Gelsomina Nardo. Calibration and validation of a terminal airport area simulator based on rotterdam-the hague airport. Master's thesis, Università degli Studi di Napoli Federico II Scuola Politecnica e delle Scienze di Base.

[38] Mina Nardo and Régis van der Sommen. Aatom - modelling, calibration, and validation an agent-based airport terminal operations model, November 2017.

[39] Mario Niazi, Muaz A; Hussain, Amir; Kolberg. Verification &Validation of Agent Based Simulations using theVOMAS (Virtual Overlay Multi-agent System) approach. *arXiv preprint arXiv:1708.02361*, pages 1–7, 2017. ISSN 16130073.

[40] NIST/SEMATECH. *e-Handbook of Statistical Methods*. Information Technology Laboratory, 2012.

[41] Craig W Reynolds. Steering behaviors for autonomous characters. In *Game developers conference*, volume 1999, pages 763–782. Citeseer, 1999.

[42] Robert G Sargent. Verification and Validation of Simalation Models. *Proceedings of the 2003 Winter Simulation Conference*, 2003.

[43] S Schlesinger. Terminology for model credibility. *SIMULATION*, 32(3):103–104, 1979. doi: 10.1177/003754977903200304. URL https://doi.org/10.1177/003754977903200304.

[44] Michael Schultz. Entwicklung eines individuenbasierten Modells zur Abbildung des Bewegungsverhaltens von Passagieren im Flughafenterminal. 2010.

[45] Jaromír Široký and H Pavlína. Optimizing process of check-in and security check at airport terminals. *MATEC Web of Conferences 236*, 2018.

[46] Dimitri P Solomatine and Avi Ostfeld. Data-driven modelling: some past experiences and new approaches. *Journal of hydroinformatics*, 10(1):3–22, 2008.

[47] Ron Sun. The importance of cognitive architectures: An analysis based on clarion. *Journal of Experimental & Theoretical Artificial Intelligence*, 19(2):159–193, 2007.

[48] Inc The MathWorks. fitdist. https://www.mathworks.com/help/stats/fitdist.html. Accessed: 2019-05-15.

[49] Arjan van den Berg. Quantifying vulnerabilities in an airport checkpoint. Master's thesis, Delft University of Technology.

[50] Paul Pao Yen Wu and Kerrie Mengersen. A review of models and model usage scenarios for an airport complex system. *Transportation Research Part A: Policy and Practice*, 47:124–140, 2013. ISSN 09658564. doi: 10.1016/j.tra.2012.10.015. URL http://dx.doi.org/10.1016/j.tra.2012.10.015.

[51] Xiaorong Xiang, Ryan Kennedy, and Gregory Madey. Verification and Validation of Agent-based Scientific Simulation Models. *Agent-Directed Simulation Conference*, pages 47–55, 2005. URL http://www.nd.edu/{~}nom/Papers/ADS019{_}Xiang.pdf.