Article

CaMS: A Monte-Carlo Runway Capacity Simulation Software

Yuntao Dai [1,†,*]

[1] College of Air Traffic Control and Management, Civil Aviation University of China, Jinbei Highway 2898, Dongli District; daiyuntao123@163.com
[*] Correspondence: daiyuntao123@163.com

Abstract: The growing demand for air transport increases the need to optimise the airspace structure and operational scheme. Airports are at the heart of air traffic flows and have long been a bottleneck in the overall system. One of the most important prerequisites is a prospective and well-designed runway capacity. Existing runway capacity software such as SIMMOD, TAAM and AirTop do not focus on runway capacity simulation. Therefore, this paper proposes a runway capacity simulation software called CaMS (Capacity Monte-carlo Simulator). A case study is conducted on ZBOW (Baotou Donghe Airport).

Keywords: Airport capacity; Capacity coverage chart; Quantile regression

Citation: Dai, Y. CaMS: A Monte-Carlo Runway Capacity Simulation Software. *Capacity Analysis of Aviation System Course* **2024**, *1*, 1–12. https://doi.org/

Received:
Revised:
Accepted:
Published:

1. Introduction

The air transport industry has developed drastically in recent years and is expected to grow by 4 per cent annually by 2040 [1]. According to a research [2], airline delays in China led to an economic loss of 42.4 billion RMB in 2016 and 72.9 billion RMB in 2018. It is therefore imperative to address the issue of delays. One of the most important prerequisites for reducing delays is to properly determine runway capacity.

Nowadays, there are several software packages that are capable of simulating runways. SIMMOD, developed by ATAC, provides the flexibility and power of true rules-based modelling capability through the innovative implementation of a generalised simulation scripting language. This greatly enhances the ability to simulate the dynamics, variability, site-specific characteristics and situation-specific factors in air traffic operations. However, it is based on FAA regulations, which limits its applicability [3]. Similar to SIMMOD, Boeing has developed Total Airport and Airspace Model (TAAM) [4]. As a gate-to-gate simulator simulator of airspace and airports. Nevertheless, its use is limited due to its poor running efficiency. With multi-agent techniques, AirTop is a new generation gate-to-gate fast time simulation software and has a realistic airport ground movements considering capture of air traffic controller roles. Yet it is not focus on runway capacity estimation [5].

To conquer the runway simulation in precision, this paper proposed a software called CaMS (Capacity Monte-carlo Simulator), which is able to establish the structure of airspace and simulate the movement on the runway focus on the runway configuration. In addition, the aircraft model are highly customizable, rendering a more realistic simulation result.

CaMS is a Java-based software whose philosophy consists of three main parts: Interface, Simulation Logic and Environment. The following sections introduce the capabilities and structure of each part.

2. Main Interface & Basic Flow

The *Main Interface* is the base of the software and displays four main panels, a menu bar and a toolbar when the software is running. As shown in Fig. 2, CaMS provides a user-friendly and intuitive user interface.

Internal classes, such as *Util.class*, *SimulationEnvironment.class*, etc., store the static or dynamic data during simulation or support running of simulation. Fig. 1 shows the structure of the software.

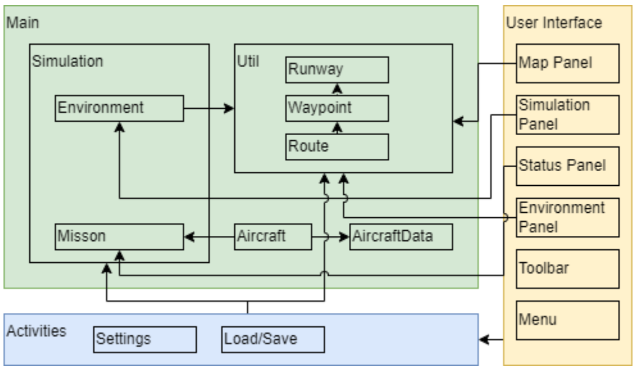
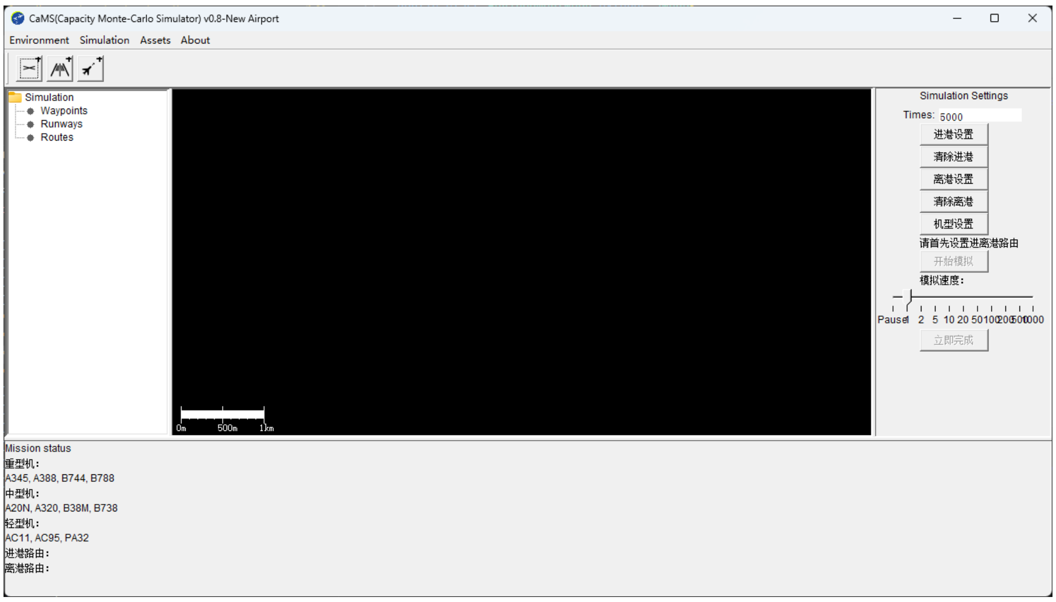


Figure 1. Schematic diagram of CaMS. Arrows demonstrate dependence of components.

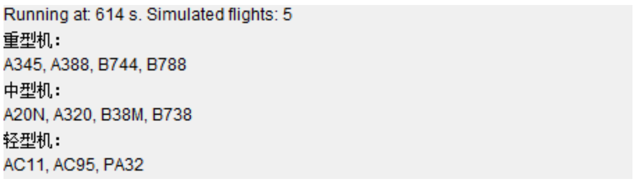


Times: 5000

请首先设置进离港路由

Mission status
重型机:
A345, A388, B744, B788
中型机:
A20N, A320, B38M, B738
轻型机:
AC11, AC95, PA32
进港路由:
离港路由:

Figure 2. Main interface of the CaMS, it displays four main panels, a menu bar and a toolbar.



Running at: 614 s. Simulated flights: 5
重型机：
A345, A388, B744, B788
中型机：
A20N, A320, B38M, B738
轻型机：
AC11, AC95, PA32

Figure 3. Status panel under simulation is running. The content of the first line is changed to the real-time iteration and simulation time.

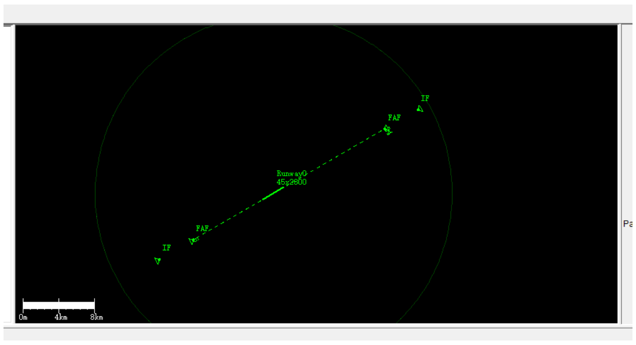


Figure 4. The map panel is displaying a simple runway layout as the user add the *Runway*0.

2.1. Main Interface & Structure

The panel on the left is the *Environment Panel*, which displays objects in the current simulation environment; the panel on the right is the *Simulation Panel*, which provides the function to configure the simulation condition, start the simulation and control the simulation speed while the simulation is running; the bottom panel is the *Status Panel*, which displays the simulation environment configuration and running status; the middle panel with black background is the *Map Panel*, which visualizes the runway configuration, airspace structure and aircraft movements.

Environment Panel is essentially a tree structure which stores the *Objects* of the simulation environment, including *Waypoints*, *Runways* and *Routes*. Specific introductions to these objects can be found in section 4.

Simulation Panel provides access to change the configuration of the simulation environment. From top to bottom one may change the simulation iteration, assign inbound and outbound routes and quickly clear the inbound and outbound route selection, select the *Aircraft* models used in the simulation, and finally start a simulation *Mission* and change the simulation speed or just run it without delay.

Status Panel displays the status of the simulation environment and the current time and iteration in real time while the simulation is running, as shown in Fig. 3.

As shown in Fig. 4,*Map Panel* is an interactive panel that not only visualizes the routes and runways, but also allows the user to intuitively add *Objects*. The left mouse button can be used to drag the view that the panel displays. The user can also scroll the mouse wheel up and down to zoom in and out as the plot scale bar in the bottom left corner changes. As shown in Fig. 5, the *Objects* manifest in the panel are colored as long as they're available: Waypoints are displayed as white dots; runways are drawn to their exact size, runway IF and FAF points are green triangular shaped and connected by a green dash line; routes are displayed in semi-opaque cyan lines; aircraft are displayed as an aircraft icon in white; aircraft information, including flight name, heading, speed and altitude, is displayed in red text.

2.2. Creating Objects

Above *Panel* zone is *Toolbar*, which mainly manipulates *Object*. Three buttons are used to add waypoints, runways and routes respectively. By clicking on one of them, the

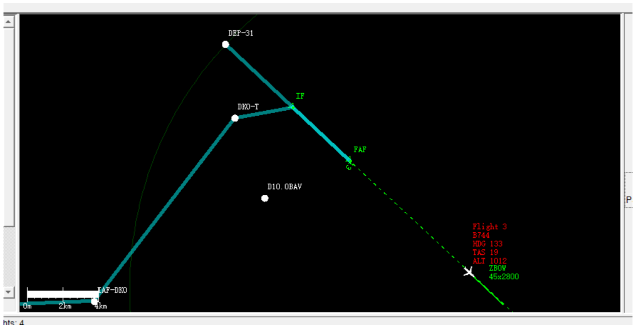


Figure 5. The map panel is displaying routes, waypoints, runway attachment and aircraft during a simulation.
Waypoints are displayed in white dots; Runways and attaching objects are displayed in green; Routes are displayed in cyan; Aircraft are displayed as an aircraft icon in white; Information of aircraft are displayed in red text.

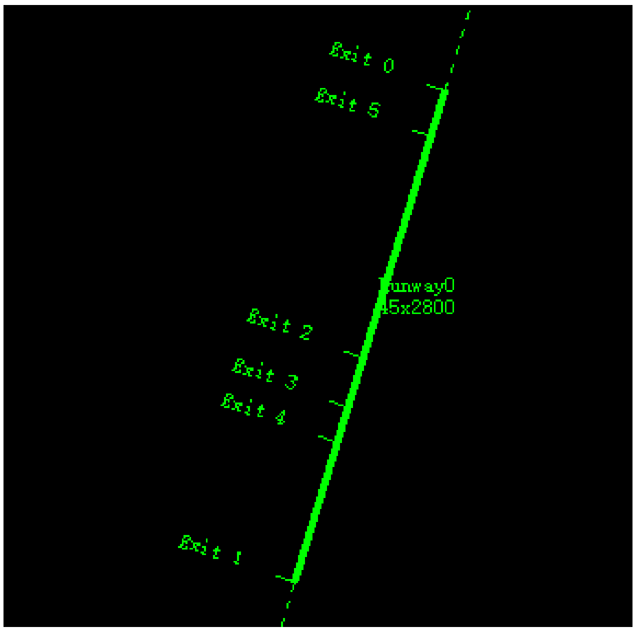


Figure 6. Exits indicators are displayed when a runway is displayed, in this case there are 6 exits on this runway.

corresponding *Object* can be added to the current simulation environment. In this case, a right click will exit the add mode and return to the default manipulation mode.

When adding a waypoint or runway, a semi-opaque layout shape will present with cursor, which means currently is in *Adding mode*. Clicking left mouse button will generate a new waypoint or runway object and exit the *Adding mode* if right mouse button is clicked instead. Otherwise it will enter *Route mode*, in which one can click on a waypoint, whether it is an individual waypoint or an attachment. This will add a waypoint to a new route waypoint sequence and display a yellow line with one end following the cursor, while another located in the previous waypoint in the sequence indicates the candidate for the next waypoint in the sequence. Right-clicking will complete the route creation process and render a new route. Note that a route must start or end with a runway FAF to be a valid inbound or outbound route.

As a result, created waypoints will display in shape of white dot besides those triangular shaped IF and FAF created by runways. The runways will render in green and for the runway itself would be drawn to scale with the scale indicated on the plotting scale bar. Alongside the runway, as shown in Fig. 6, there may be some exit indicators to mark the exit of runway, they may be of diverse forms, for example, 90-degree exit and U-turn exit.

There are also several lines of text describing runway characteristics, including runway name and size. Extended lines of runways are dashed and end at IF point. Waypoints are connected in route order to show the routes, which are shown as semi-opaque bold cyan lines.

2.3. Menu

Menu bar has four terms, *Environment*, *Simulation*, *Assets*, and *About*. *About* merely displays the version of the software and other three menus have functionalities to actual simulation. *Environment* have the capability of loading and saving the environment. Specific information may found in section 3. *Simulation* may completely erase all data, including running and static data. *Assets* has the same function as the *Toolbar*.

2.4. Simulation Execution

A simulation may initiated by clicking the Start button on *SimulationPanel* once the simulation time, inbound and outbound routes have been set. This will enable the manipulation on simulation speed slider which is 1x by default. Drag the slider may change the simulation speed up to 1000x or pause the simulation. For a faster simulation, one may click on finish instantly, which will dispose the timer and make the simulation run by thread.

The simulation result will be saved to a .csv file named 'Out_' and execution time in the path where the software is executed.

3. Simulation Environment

Environment implemented by *SimulationEnvironment.class*.The simulations may run in form of *Mission* in current simulation *Environment*. A new *Environment* will be created when open the software. One may save or load the *Environment* by menu.

This class stores all the waypoints, runways, and routes, along with inbound and outbound information. Apart from that, it also contains aircraft in simulation.

3.1. Utility

Utility implemented by *Util.class*. The class statically stores the current simulation environment and provide various of interfaces that access to main window, map panel, environment tree, simulation settings, etc. Mouse position for calculating drawing offset and manipulate objects is also stored here.

Additionally, it provides the unit conversion tools, such as knots to meters per second, nautical miles to meters, etc., and some check method, like exit check and separation check.

3.2. Separation Standard

Separation standard of CaMS is from *Management Regulations for Civil Aviation Air Traffic* by CAAC. In Chapter 6, Section 5, it regulates the wake flow separation standard, as shown in Table 1 and Table 2.

Leading \Trailing	Heavy Aircraft (non-A380-800)	Medium Aircraft	Light Aircraft
A380-800	11.1	13.0	14.8
Heavy Aircraft	7.4	9.3	11.1
Medium Aircraft	–	–	9.3

Table 1. Wake separation standards by CAAC (in distance).

Trailing\ Leading	A380-800	Heavy Aircraft	Medium Aircraft
Medium Aircraft	3	2	–
Light Aircraft	3	2	–

Table 2. Non-Radar separation minimum tail clearances by CAAC(in minutes)

3.3. Save or Load Environment

One may save simulation environment or load environment by *Environment* menu. Simulation evironment is saved as .xml file, it contains all waypoints, runways and routes in specific simulation environment.

3.4. Objects

There are three types of objects in environment, waypoints, runways, and routes. They are created by manipulating map panel that section 2 have been introduced.

Waypoints is an array of $Waypoint$, the class $Waypoint$ records the coordinate, type and name of a waypoint. In addition some navigation restrictions such as direction and height are also listed in the class. Further, the variable *belongsto* indicates whether a waypoint is attach to a runway.

Runways is an array of *Runway*, the class *Runway* records many properties of runway. Coordinate, angle and size are physical and basic properties of a *Runway*. However, to support inbound and outbound simulation, the class extends the MDA and IF assignment, one may set the relative position of IF and FAF along the runway extend line. In addition, precise simulation need exact time that aircraft spent on runway, hence the runway exit information, including relative position to datum point and geometric shape are listed in the class. The availability of runway is also considered.

Routes is an array of *Route*, the class *Route* records the waypoint sequence of a route. The preceding aircraft and its initiate time are also recorded here for separation check.

4. Simulation Logic

With the simulation environment established, start a simulation will generate a $Mission$. The mission will terminate when the number of simulated flights is satisfied.

The mission start with generate a output file. Then it will constantly randomly generate $Aircraft$ is of various aircraft models whose performance models are stored in $com.cams.aircraftdata$. Depends on route type and navigation status, aircraft may landing at or takeoff from a runway. A flight will terminate and vanish on screen when it reaches the last waypoint in route or landing procedure is finished.

4.1. Aircraft & Aircraft Performance Data

$Aircraft$ is the core of the software, to precisely simulate aircraft behavior, there are series of methods and variable are applied. Fundamentally, class $Aircraft$ stores the type and name of the aircraft. When simulation is running, its coordinate, speed and heading will change accordingly, they also recorded in the class.

An aircraft basically has three types of maneuver. Navigation, landing and takeoff. Therefore the class has navigate, landing and takeoff methods to simulate it. Navigation is merely ensure the heading, altitude and speed are in the right range. Landing and takeoff, by contrast, need to change the behavior during the process, for example, the aircraft would level and its vertical speed would slightly descend seconds ahead touchdown.

Specific model of aircraft extends from abstract class $AircraftData$. The performance data stored various flight parameter in seven stages of a flight, including takeoff, initial climbing, mach climbing, cruise, initial descending, descending and landing. Additionally with the properties of MTOW, name, category, etc.

In takeoff stage, there are takeoff distance, V_2 and rotate speed V_R, where V_R is calculated by takeoff distance and V2. After takeoff, initial climbing has its own ROC and IAS. Similar to initial climbing, other climbing/descending stages have their own ROC or ROD, and IAS or TAS likewise. At cruise stage, the model has cruise mach and ceiling altitude. At last, landing stage modeled IAS, ROD touchdown speedV_{at} and landing distance.

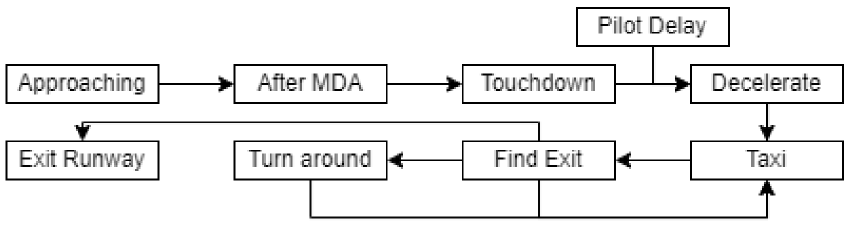


Figure 7. Schematic of possibly landing procedure.

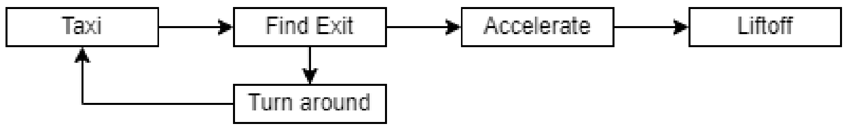


Figure 8. Schematic of possibly takeoff procedure.

4.2. Inbound Routes & Landing

An inbound route starts with a normal waypoint and ends at runway. Actually, in internal logic, the navigation ends at aircraft reached FAF. After FAF, the aircraft changes into approaching mode.

As shown in Fig, 7, approaching aircraft will continue faster descending until MDA, after MDA the aircraft will gradually level and float down toward runway. After touching down, a delay is introduced before deceleration due to reaction delay in pilots. When aircraft decelerates to a proper speed, it will either exit runway or accordingly turn around then exit which triggers continually taxi on runway. The whole landing procedure ends at aircraft have found the final exits.

4.3. Takeoff & Outbound Routes

An outbound route starts at runway entrance and ends at the last waypoint of the route. The aircraft will randomly appears on taxiway and find possible runway entrance for takeoff. Subsequently, it enters runway either do a takeoff or accordingly turn around then takeoff which triggers continually taxi on runway. The whole takeoff procedure ends at aircraft reach the position of FAF.

5. Case Study

The simulation is run under the scenario that routes are evenly distributed and based on ZBOW(Baotou Donghe Airport). As shown in Fig. 9, ZBOW is a one-runway airport. In CaMS, it can be modeled as Fig. 10. Compares to official approach chart Fig. 11 and Fig. 12, the model structure is of high similarity.

Partial result is listed in Table 3. The envelope and bubble plot of the airport is demonstrated in Fig. 13 and Fig. 14. The proportion of exit usage is shown in Fig. 15. It's obvious that light aircraft exit earlier and exit 2 is the most frequently used exit.

References

1. Boeing. *Boeing Current Market Outlook 2020-2039*; Boeing Company, 2020.
2. J.W., Z.J.H.; Y.Y, Q.; L.L., W. Airline Delay Cost Estimation Based on Trans-log Cost Function. *Logistics Sci-Tech* **2020**, *43*, 35–38,45.
3. Rakas, J.; Mumayiz, S. Airport-Airspace simulations for capacity evaluation. *Transpostation Research Circular, n. E-C035* **2001**.
4. Sood.; Wieland. Total airport and airspace model (TAAM) parallelization combining sequential and parallel algorithms for performance enhancement. In Proceedings of the Proceedings of the 2003 Winter Simulation Conference, 2003. IEEE, 2003, Vol. 2, pp. 1650–1655.
5. Varun, D. Estimation of runway capacity at Changi Airport using AirTOp and Monte Carlo analysis. PhD thesis, 2015.

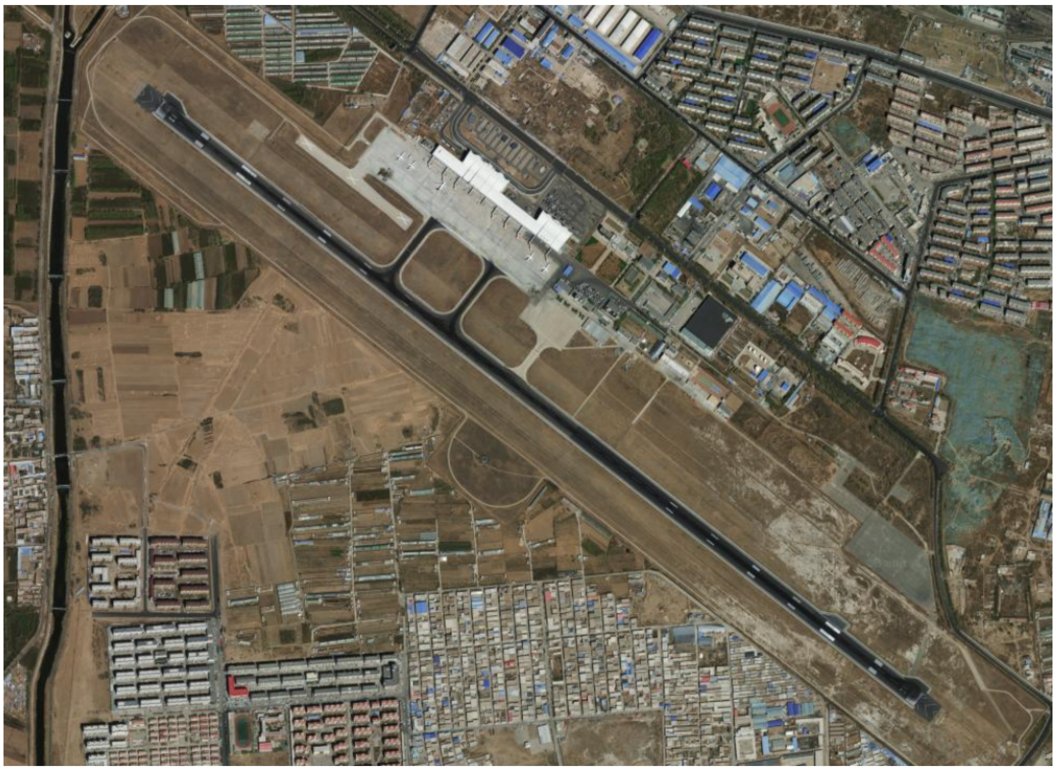


Figure 9. Satellite picture of ZBOW.

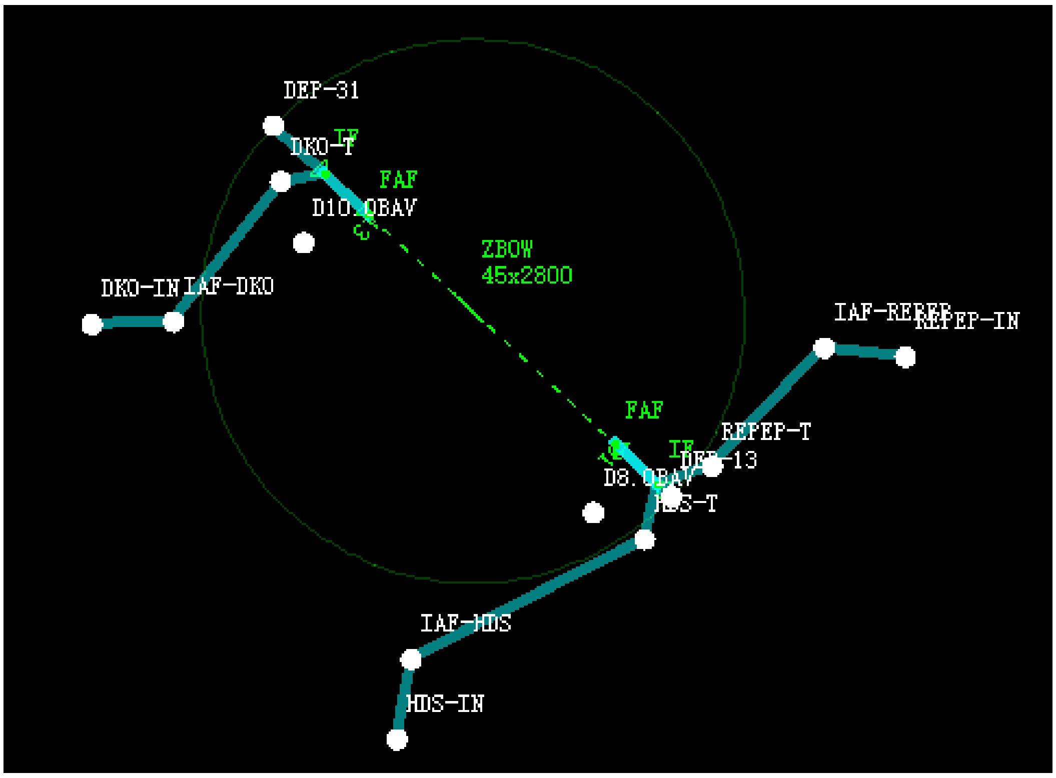


Figure 10. Airspace configuration of ZBOW, i.e. simulation environment.

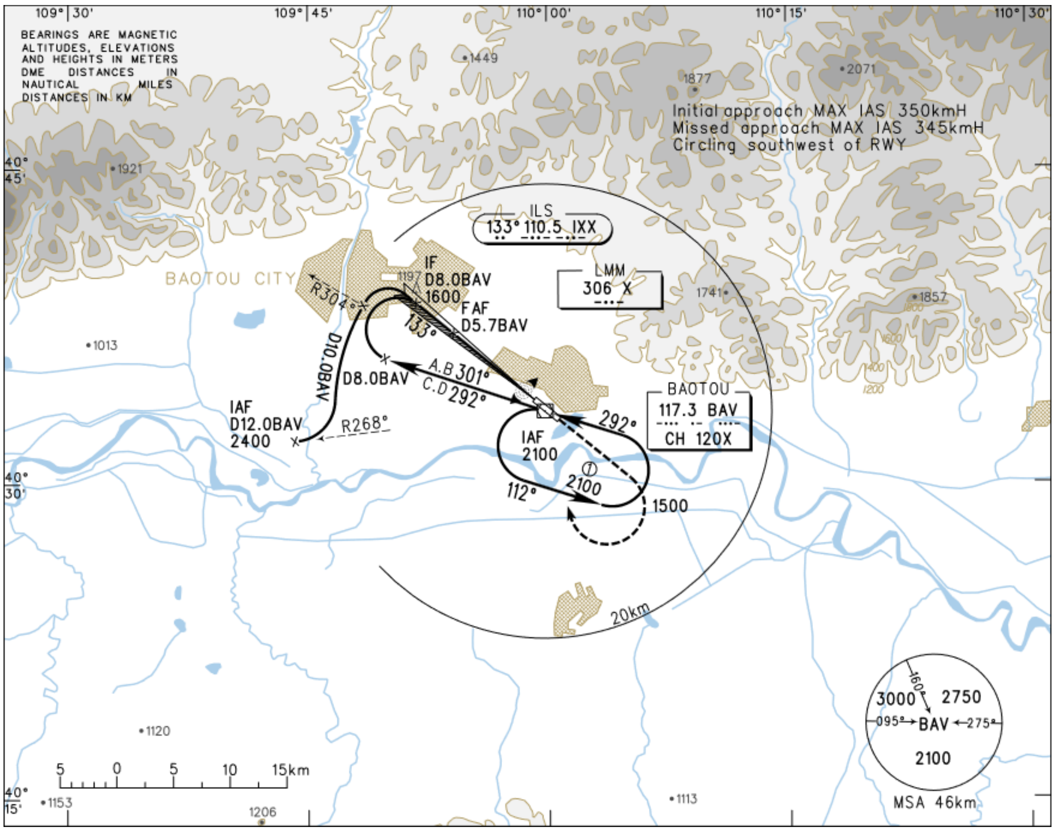


Figure 11. Instrument approach chart for runway 13.

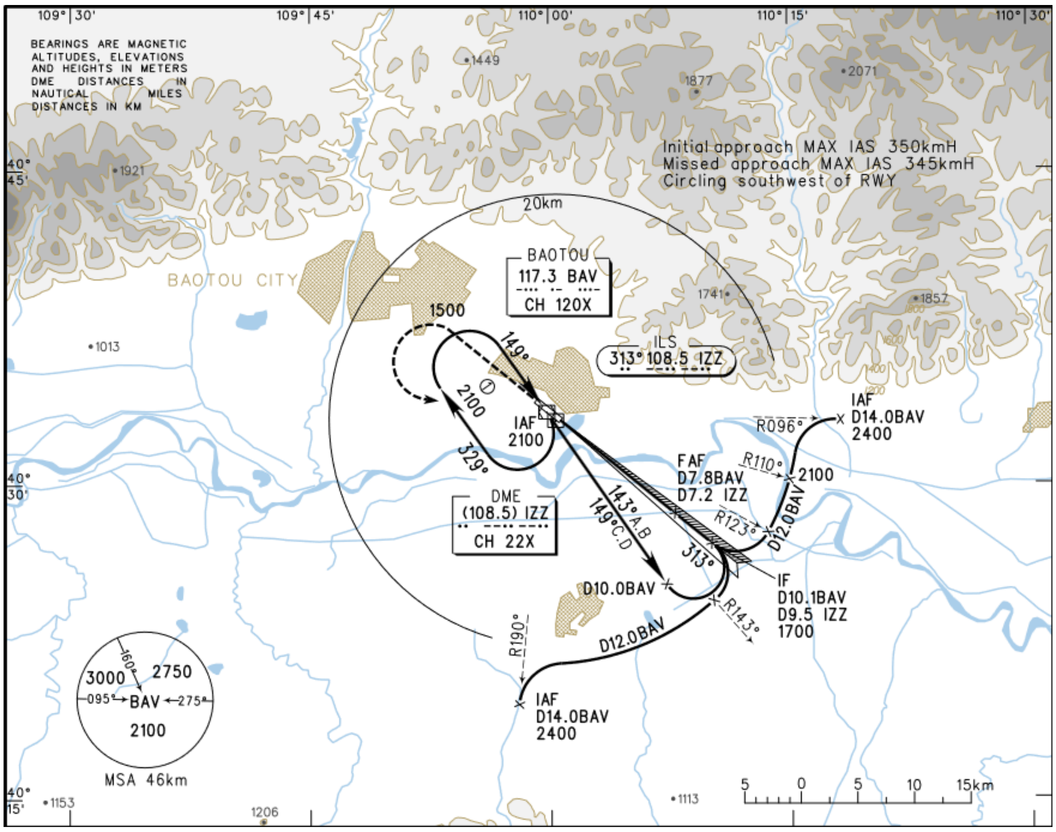


Figure 12. Instrument approach chart for runway 31.

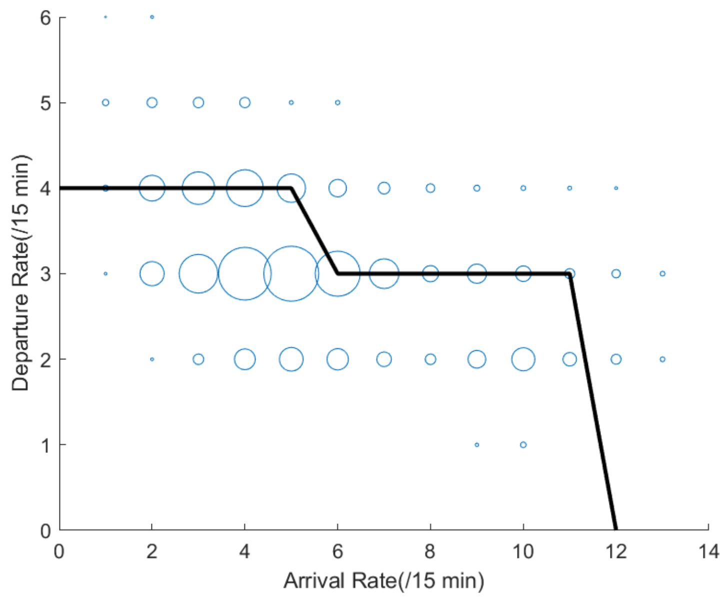


Figure 13. Bubble and envelope plot during 15 minutes.

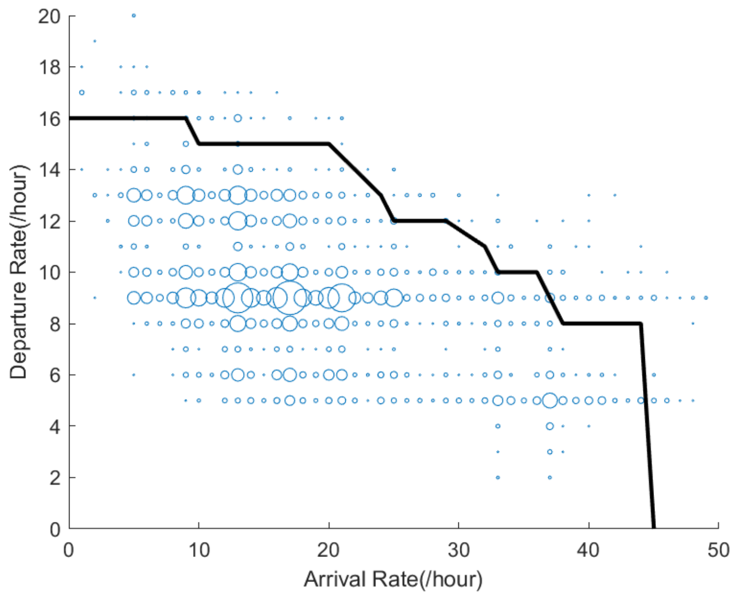


Figure 14. Bubble and envelope plot during 1 hour.

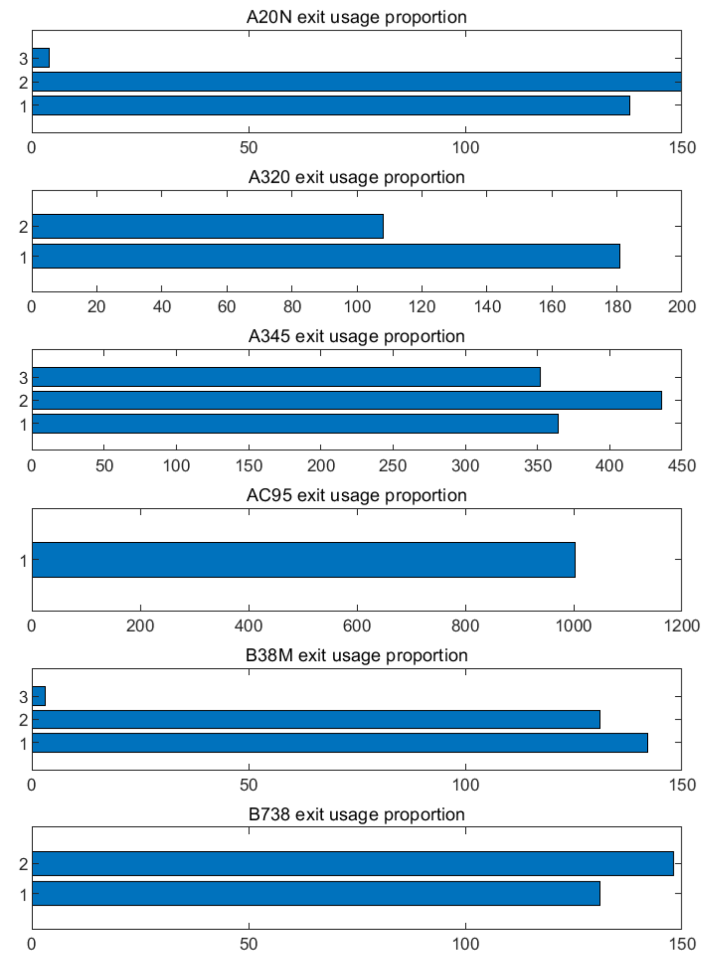


Figure 15. Diverse exit usage by different flight models.

Time	Flight	Type	Event	Exit
6456	Flight 54	A345	Exit	4
6615	Flight 60	B38M	Enter	0
6679	Flight 60	B38M	Liftoff	–
6756	Flight 58	A345	Touchdown	–
6848	Flight 57	A345	Touchdown	–
6892	Flight 57	A345	Exit	3
6929	Flight 59	AC95	Touchdown	–
6930	Flight 58	A345	Exit	2
6974	Flight 59	AC95	Exit	2
7036	Flight 64	A320	Enter	0
7092	Flight 64	A320	Liftoff	–
7245	Flight 61	B38M	Touchdown	–
7285	Flight 61	B38M	Exit	2

Table 3. Partial data in output csv file.