Abstract

This project constructs a framework that automatically reproduces 3D flight trajectories from multiple bat call positions. For reconstructing 3D trajectories from 4D points, segments consistent with a biological model are combined together, with larger segments consistent with the biological model selected to be the subset of all possible trajectories. In each subset, the trajectory with the longest length or the highest likelihood is selected and output.

This project comprised three stages. Firstly, biological flight priors of the bat *Nyctalus leisleri* are constructed by filtering out outliers from statistical measurement models. These constructed models of previous flights are integrated with a model based filter using a maximum likelihood method that pre-selects those trajectories with higher probability.

Next a trajectory's segment subset selection algorithm is generated. Average linkage agglomerative clustering is utilized both at the beginning and at the end of this algorithm. In each clustering, bifurcation checks help to measure the distance between each flight trajectory, and average linkage is utilized to calculate the similarity between the clusters. Additionally, an automatic minimum threshold selection method, used for dividing the dendrogram, is proposed here. After the first clustering, a branch-and-bound algorithm is implemented to help to extend the branches of each template trajectory's segment subset. Its associated boundary and pruning rule references the results of the maximum likelihood method once normalized and compared with its neighbours.

Speed optimization is considered. Two methodologies are proposed to solve the crucial problem of a large operation time. A selection method using adaptive interpolation points is proposed to achieve both a high speed and a high resolution interpolation calculation in curvature measurement; additionally, an importance sampling method is implemented by looking at a 3 point segments table to achieve the most efficient branching.

Finally, three assessment methods are proposed here to obtain an objective and reliable evaluation: reconstruction rate for bat trajectories; recognition rate for bat call positions; and the average unbiased distance deviation of each trajectory.

This research provides both a feasible method of recognizing bat flight trajectories and a repeatable way of data analysis independent of subjective manual observation.

Table of Contents

List of Figures

List of Tables

1. Introduction

Bat research is not only of interest for biologists, but also for the fields of agriculture, military, and industry. Flight and echolocation are unique features of bats that separate them from other mammals and have facilitated their range and diversity of habitats. Thus, it is valuable to record both the flight paths and vocalizations of these creatures to gain a complete understanding of their behaviour. There are shortcomings in current research when analyzing bat flight trajectories. For example, video source research often finds its field of vision is largely restricted by occlusions such as trees and the narrow field of view captured by the camera compared to acoustic tracking. On the other hand, for audio source research, often an excess of valuable manpower is spent in recognizing and reconstructing correct bat flight trajectories, especially when analyzing groups of bats. Therefore, the aim of this paper is to design and implement a framework that can automatically reconstruct multiple flight trajectories from bats' call positions thus improving current research methods. The research undertaken is divided into two parts: bat behaviour modelling and analysis of flight path recognition.

There has been much research on the subject of bats. Some researchers have focused on the relationship between bat species and specific behaviours such as flight speed (Hayward and Davis, 1964; Whitaker et al., 1993; Shiel et al., 1999; Riskin, 2001; Hopkins et al., 2003; Russ et al., 2003; Russo et al., 2005; Sánchez-Hernández et al., 2006; Akins et al., 2007; Grodzinski et al., 2009; Obrist et al., 2011). A number of researchers have analyzed the relationship between bat species, echolocation, and their trajectories (Denzinger et al., 2001; Russo and Jones, 2002; Jensen et al., 2005; Hiryu et al., 2007; Surlykke et al., 2009; Goerlitz et al., 2010; Moss and Surlykke, 2010; Corcoran et al., 2011). Other researchers have focused on how to precisely record bats' flight features like 2D/3D flight trajectory, flight velocity/acceleration and sonar spectrum characteristics by camera and acoustic sensor arrays (Erwin et al., 2001; Ghose et al., 2001; Ghose and Moss, 2003; Lee et al., 2004; Eastman and Simmons, 2005; Theriault et al., 2010).

In contrast, this project aims to use 3D raw spatial-temporal data from echolocation sounds with prior information on flight features to build a model that represents possible bat flight paths. To the model is then applied to reconstruct/detect flight paths using three basic elements of pattern clustering, smoothing, and branch-and-bound methods. Additionally, two optimization methods utilizing computer graphics theories are proposed here for solving the most crucial problem, the large computational time. Unlike most online vision tracking, this project is focused on off-line trajectory reconstruction.

An overview of the process is shown in figure 1. Pre-processing includes a coordinate system converter and a temporal and spatial data points rectifier. The former converts raw data into a WNK file represented by spherical polar coordinates to Cartesian coordinates, and the latter implements two channel unbalance checking, temporal resolution checking, sorting and monotone constraining, and physical reliability checking. These pre-processed raw data points are fully connected with each other and feed into the first clustering in the main algorithm – the trajectory's segment subset selection algorithm. In this algorithm, all segments will be clustered into several small groups; in each group, segments will be branched to extend their length in each iteration of this algorithm.

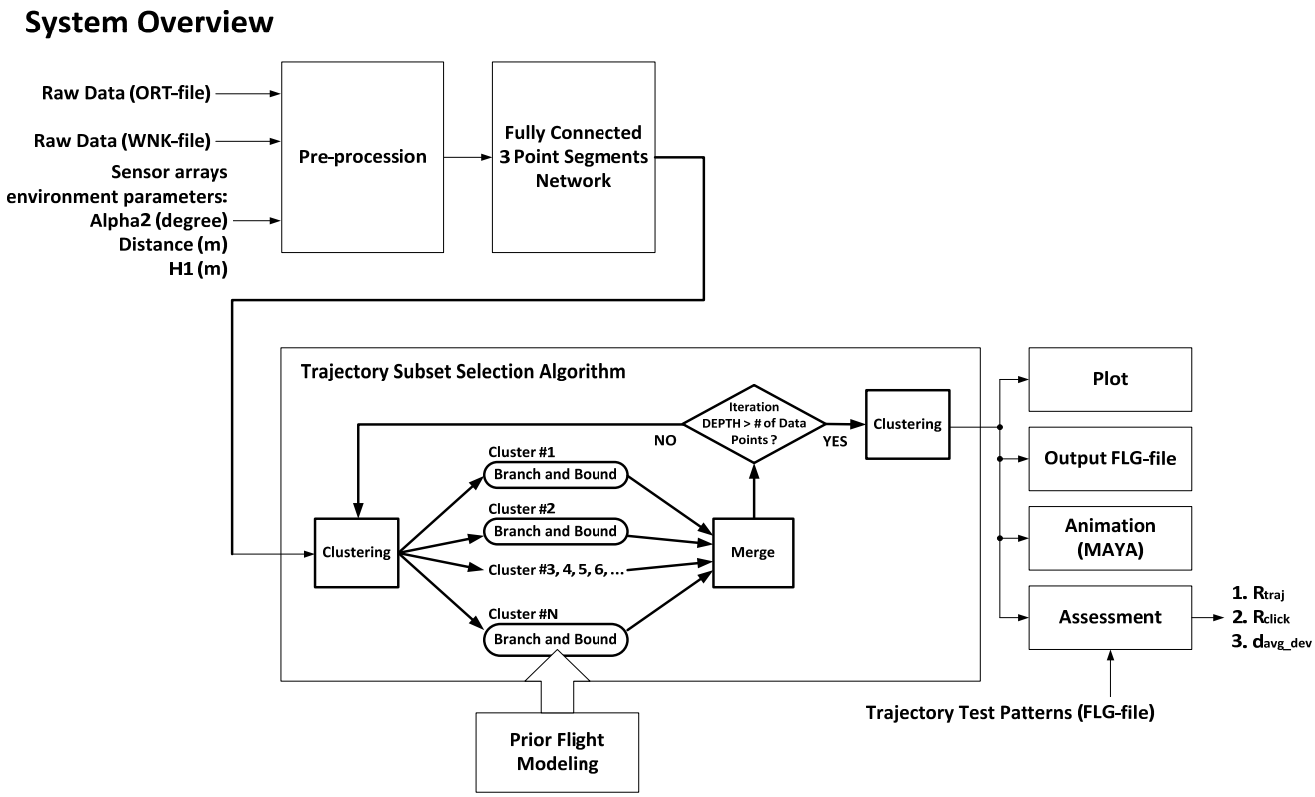


Figure 1. System overview

Prior biological models are constructed by filtering out outliers from measurement models which have captured over six flight features of the bat *Nyctalus leisleri* by training hundreds of raw data files. These prior models work as filters and are combined with the function block - branch-and-bound - in the main algorithm. In branch-and bound, the flight features of each smoothed trajectory will be compared with prior models. If the flight features of one trajectory bear greater similarity than others to prior flight models, then such a trajectory will be given permission to extend its length. After the final clustering, the trajectory with the longest length or the highest likelihood in each cluster will be selected and output.

Output trajectory patterns will be plotted by Matlab, written to FLG files, fed into MAYA for animated presentation, and then assessed by comparing them with trajectory test patterns. These trajectory test patterns have been reconstructed by the biologist manually. Three different assessment methods are proposed here to gain an objective and reliable evaluation.

2. Background

2.1 Target Domain: Bat Flight

This section contains relevant background information on bats.

2.1.1 Echolocation

Early bat research in biology focused on the classification of species and the relationship between species, reproduction, migration and distribution, feeding habits, and flight behaviour. Prior research on the importance of bats and their close link to the biological food chain and agriculture has provided important fundamental knowledge.

From a traffic, industry and military perspective, the most important discovery in bat research has been echolocation (Griffin, 1958). This exploration has not only expanded the area of bat research itself, but has also driven the progress of acoustics and its application, and has enabled humans to gain a large variety of knowledge from this most diverse group of mammals. For example, people have investigated bats utilizing morphological, physiological and biochemical techniques, and applied mechanics and aerodynamic theory to analyze and have subsequently learned how to improve flight efficiency and increase flight capacity (Maina, 2000; Watts et al., 2001). Moreover, the principles of object detection and classification by ultrasound learned from bats have been applied to radar and sonar systems. Because their capabilities have evolved for over 50 million years, studying or imitating bats has provided an excellent way to form a robust, adaptive and accurate detection system (Holderied et al., 2008; Hurtado and Nehorai, 2008; Balleri et al., 2009, 2010[a]; 2010[b]).

To facilitate movement in the dark, bats have developed echolocation behaviour. Ultrasound is emitted from bat's mouth or nose, reflected off objects like walls, trees or insects and received back by their ears. Therefore, echolocation enables bats to recognize an object's position, size, direction of movement, velocity and even its texture.

The sounds of a bat's echolocation signal can be broadly divided into two categories, constant frequency (CF) and frequency modulated (FM). The signal characteristic of the constant frequency is simple, and it has a lower bandwidth and longer duration. In contrast, the frequency modulated signal is more complicated: it has a broad bandwidth, a short duration time and is often multi-harmonic. Each sound category has its own merit; Simmons et al. (1979) and Bell and Fenton (1984) have all stated that the constant frequency method

has a better ability to assess the relative velocity between the target object and itself because of the Doppler Effect. In contrast, the modulated frequency signal is preferred to recognize the distance, direction, and texture information from the target object. In reality, these two techniques rarely occur in isolation. In fact bats' echolocation can be divided into three different methods: short-CF/FM, FM/short-CF, and long-CF/FM, learned from the features of their sound emission patterns (Simmons et al., 1979). We know that each species of bat has its own distinct acoustic signal type that will be used in echolocation, and which can also be varied in a situation-dependent manner.

Besides the potential of sound emission patterns of different species being different, within the same species the pulse repetition rates of sound emission patterns, frequency, and call duration will also be change during insect capture. In general, the rate is slow before the bat finds a prey, but this rate will be increased as the bat detects and approaches its prey to keep abreast of the movement and position of the prey until its capture. This foraging sequence can be divided into three phases: search, approach, and "terminal buzz", as figure 2 illustrates (Moss and Surlykke, 2010). The pulse repetition rate is about 5 to 10 calls per second in the search phase, and increases from 20 to 80 to 170 calls per second in the approach and final phases.

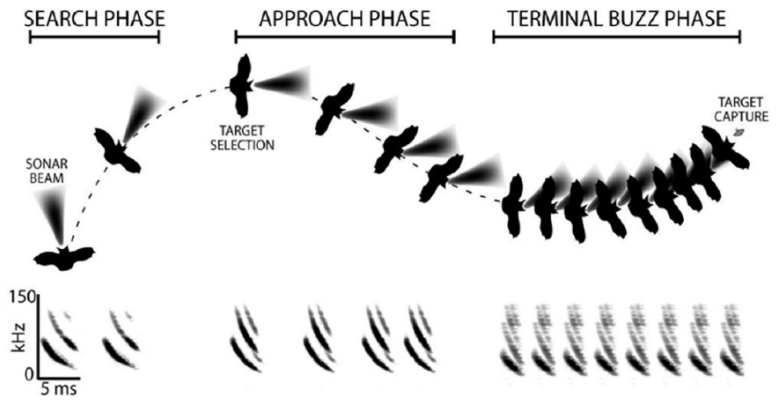


Figure 2. The change of pulse repetition rate in a bat's foraging sequence
Source: Moss and Surlykke, 2010.

Based on these features of echolocation, the biologist can collect bat data and classify them much more easily than a spot watch. This technique was first used to collect bat distribution data in the London area in 1965 until 1980 (Hooper et al., 1981). Today, such use of ultrasonic receivers and the monitoring of the echolocation of bats has been a main trend in bat research. Of course, such audio source localization has its own drawback, and this will be described in detail later.

Alternatively, video source localization is another useful method to collect bat data such as trajectory. The issue here is the considerably restricted vision by occlusions such as trees; the comparatively small field of view, and poor illumination levels. Moreover, gaining azimuth and elevation information to construct a three dimensional flight trajectory is another big task.

2.1.2 Bat Groups

It is well known that bats aggregate into sleeping groups for thermoregulation (Twente, 1955, 1956; Herreid, 1963). Bats may gather in clusters with body contact comprising from a few to thousands of individuals (Allee, 1927), which then fly out together for foraging in the daytime. This interesting aggregation behaviour provides an excellent research topic for the fields of animal aggregation and group behaviour, such as emergency behaviour (Allee, 1927; Mills et al., 1975). Nevertheless, it complicates the work of bat monitoring whether using video or audio source localizations in open fields.

Indoor experiments are much simpler than field experiments. For example the parameters in the former can be controlled much better e.g. bat species, bat numbers, wind speed, volume of space and illumination. However, their results are usually restricted, not universally applicable, and are not applicable for all conditions because bats change their call parameters and flight behaviour according to their surroundings (Hagino et al., 2007; Hayward and Davis, 1964). Indoor experiments can only replicate similar situations, such as bats flying in a mine shaft or cave tunnel (Lee et al., 2004).

These facts reveal the importance of a robust methodology that can cope with most environmental and measurement problems, and one that can also collect and classify data correctly.

Table 1. Previous work whose experimental places are located in open fields

Studies	Bat species (Indoor/Field studies)	Experiment methods
Eastman and Simmons (2005)	*Eptesicus*, *Myotis* and *Pipistrellus* (Field)	1 camera and 1 microphone array
Goerlitz et al. (2010)	*Nyctalus leisleri* and *B. barbastellus* (Field)	microphone arrays
Hagino et al. (2007)	*Pipistrellus abramus* (Field)	microphone arrays
Lee et al. (2004)	*Tadarida brasiliensis* (Field)	2 cameras
Theriault et al. (2010)	*Tadarida brasiliensis* (Field)	3 cameras
Zheng et al. (2009)	*Tadarida brasiliensis* (Field)	3 cameras

Table 1 summarizes some bat field studies in recent years. Their monitoring methodologies can be divided into three types: audio, video, and hybrid type. The type of audio (using microphones or microphone arrays) utilizes acoustic theory to record precise call emission positions as shown in figure 3. Each set of sensor arrays returns one 3D position for a bat call. For higher accuracy, positions from more sensors or sensor arrays can be combined (Hagino et al., 2007; Grodzinski et al., 2009; Goerlitz et al., 2010). However, errors introduced by echo interference and grouping delay remain problematical and cannot be fixed. Besides, the reconstruction of bat path is still by manual means, compared to the nearest previous study. Of course, this is not satisfactory in terms of efficiency and accuracy, especially for measurements of bats in groups.

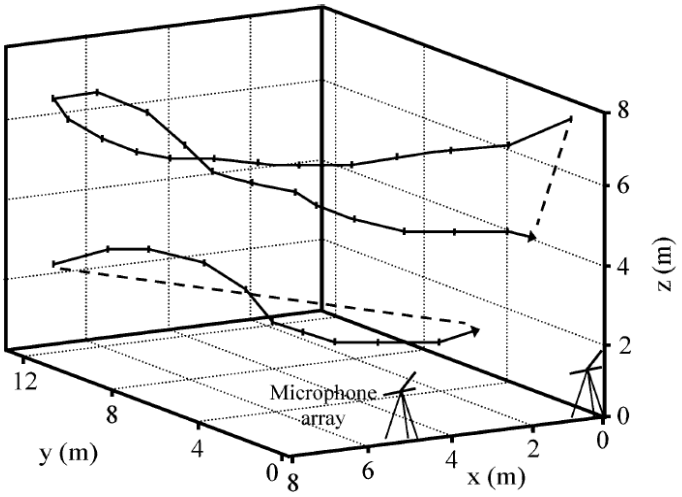


Figure 3. Example of flight trajectory monitoring from two microphone arrays
Source: Grodzinski et al., 2009.

The ambiguity caused by occlusion in single camera monitoring can be solved by using several cameras at the same time. So, theoretically speaking, the video camera type can help the biologist to explore the behaviour of animals in three dimensions effectively and efficiently. After preparation such as camera synchronization and compensation, and the extraction of the focal length and extrinsic parameters for each camera, some computer vision algorithms are utilized. For example, epipolar lines may be created by the projection of 2D points in the single view, and then 3D points can be reconstructed (Lee et al., 2004; Zheng et al. 2009; Theriault et al., 2010). Besides some fundamental problems that can occur in the computer vision area such as illumination and background noise; cross-view problems; and the inherent noise from camera, another realistic problem for researchers is the expense of equipment such as thermal infrared or high-speed cameras. Moreover, camera compensation and the optimal choice of look-at point of camera in the preparation are very important factors that can affect the experiment's result directly.

Finally, hybrid types integrate not only the positions provided by video, but also the azimuth and elevation information provided from acoustic data. This methodology appears the perfect method that can cope with most problems that occur in the two methods in isolation, nevertheless an issue that arises is in the actual combination. There are lots of potential problems that can happen in this phase. In fact some commentators feel that little merit can be found in this method (Eastman and Simmons, 2005). Moreover, similar to the issues with the audio analysis, the automatic reconstruction of a 3D trajectory is also hard to implement this hybrid type of investigation. Most previous works exclude unwanted call positions and also connected agreeable points in sequence manually.

2.1.3 Bat Trajectory

An effective bat flight path includes accurate positional and temporal information. This is also known as its flight trajectory and it helps the biologist to explore bat species. Realistically, it is hard to identify a bat species from the trajectory alone because all trajectories are so similar, but flight characteristics such as speed; acceleration; height and tortuosity information; and flight behaviours such as target capture behaviour - including detection, localization, tracking, and interception of insect prey (Erwin et al., 2001); obstacle avoidance; landing flight;, and commuting flight (e.g. flying through a cave or tunnel, or in an open field) all contribute to analysis. Besides, inter-individual and group behaviour can also be observed, such as mutual adaptive behaviour; call changes; and flight path changes between the bats. Therefore, these flight behaviours, strategies and their relationship with vocal control can be quantified and analyzed more easily (Surlykke et al., 2009).

Table 2 describes some previous work where bats' 2D or 3D trajectories were reconstructed in their studies. Of these prior studies, most have been unable to reconstruct trajectory automatically, except the studies from Theriault et al. (2010) and Zheng et al. (2009).

Therefore, this is why this research is attempting to design and implement a framework that can automatically reconstruct multiple flight trajectories from bats' call positions. Indeed, it may be the first report that attempts to solve this problem using an audio tracking method.

Table 2. Previous works those experiment places are located in open fields

Studies	Bat species (Indoor/Field studies)	simultaneously tracked individuals	Experiment methods
Corcoran et al. (2011)	*Eptesicus fuscus* (Indoor)	1	2 cameras and 1 microphone
Eastman and Simmons	*Eptesicus*, *Myotis and Pipistrellus*	1	1 camera and 1 microphone array
Erwin et al. (2001)	*Eptesicus fuscus* (Indoor)	1	2 cameras and 2 microphones
Ghose et al. (2001)	*Eptesicus fuscus* (Indoor)	1	2 cameras and 2 microphone arrays
Goerlitz et al. (2010)	*Nyctalus leisleri* and *B. barbastellus*	>1	microphone arrays
Hagino et al. (2007)	*Pipistrellus abramu*s (Field)	1	microphone arrays
Hiryu et al. (2007)	*Pipistrellus abramus* (Indoor)	1	2 cameras and 1 microphone
Jensen et al. (2005)	*Eptesicus fuscus* (Indoor)	1	2 cameras and 2 microphones
Lee et al.(2004)	*Tadarida brasiliensis* (Field)	1	2 cameras
Surlykke et al. (2009)	*Eptesicus fuscus* (Indoor)	1	2 cameras and 3 microphone arrays
Theriault et al. (2010)	*Tadarida brasiliensis* (Field)	Crowd	3 cameras
Zheng et al. (2009)	*Tadarida brasiliensis* (Field)	Crowd	3 cameras

2.1.4 Bat Trajectory Modelling

Flight trajectory encompasses important bat flight information such as flight speed, acceleration, height, tortuosity, flight behaviours, and group behaviours. We can employ the quantitative data collected from one monitoring type (audio, video, hybrid or just from a spot watch) to be the trial data for implementing a machine learning algorithm that can classify bat species and their flight paths from raw data.

Firstly, a definition of “trajectory” is essential. The spatial-temporal bat trajectory in this project is a 3D time series data of the notation:

$$\{x(t), y(t), z(t)\}_{t=0}^{T}$$

where (x(t), y(t), z(t)) are the coordinates of a bat detected at time t (Yang and Chen, 2010). We can use these coordinates to gain other important measurements, like speed, acceleration and curvature radius from segment trajectories. Furthermore, Table 3 shows trajectory parameters published by other studies. In the tables, flight speeds and turning diameters from seven major species provide clear profiles that can be included in our algorithm, for example a filter can be implemented before clustering. Additional data

acquired by Dr. Holger R. Goerlitz will be analysed later.

Moreover, Table 3 also reveals some important information: flight speed in commuting is faster than in foraging because the latter incorporates more turns; a larger turning diameter brings a faster flight speed; and there is little speed difference between intimate species. Of course, there are other factors that can affect flight speed which have not revealed in this table, such as heading direction. For example, the speed of bats heading away from the roost is faster than those heading in the general direction of the roost (Grodzinski et al., 2009).

Table 3. Trajectory parameters comparison by species

Bat species (task)	Speed (m/s)			Turning diameter (m)	Studies
	Min	Max	Mean		
Nyctalus noctula (Commuting)	8.33	15.00	11.67		Gaisler et al. (1979)
Nyctalus leisleri (Commuting)	8.89				Shiel et al. (1999)
Nyctalus leisleri			10.3		Goerlitz et al. (2010)
Nyctalus leisleri (Foraging)			6.59	2.42	Obrist et al. (2011)
P. pipistrellus (Foraging)			4.27	1.02	Obrist et al. (2011)
P. pygmaeus (Foraging)			4.49	1.13	Obrist et al. (2011)
P. nathusii (Foraging)			4.70	1.23	Obrist et al. (2011)
P. kuhlii (Foraging)			4.37	1.07	Obrist et al. (2011)
Myotis lucifugus	2.24	8.49	5.36		Mueller and Emlen (1957)
Myotis keeni	3.13	4.34	3.73		Hayward and Davi (1964)
Myotis thysanodes	3.31	4.38	3.84		Hayward and Davi (1964)
Myotis velifer	3.40	5.72	4.56		Hayward and Davi (1964)
Myotis volans	4.16	4.74	4.43		Hayward and Davi (1964)
Myotis yumanensis	2.95	3.98	3.46		Hayward and Davi (1964)
Myotis daubentoni (Foraging)			3.97	0.88	Obrist et al. (2011)
Eptesicus fuscus	3.58	6.93	5.25		Hayward and Davi (1964)
Eptesicus serotinus (Foraging)			5.24	1.53	Obrist et al. (2011)
Eptesicus nilssoni (Foraging)			4.27	1.01	Obrist et al. (2011)
Tadarida brasiliensis	3.08	4.74	3.91		Hayward and Davi (1964)
Tadarida femorosacca	3.58	4.56	3.98		Hayward and Davi (1964)
Tadarida molossa	3.93	5.14	4.54		Hayward and Davi (1964)
Tadarida teniotis (Foraging)			5.07	1.44	Obrist et al. (2011)
Tadarida brasiliensis (Commuting)	4.17	12.50	8.33		Williams et al. (1973)
Tadarida brasiliensis (Commuting)			9.38		Theriault et al. (2010)
Hypsugo savii (Foraging)			4.25	1.01	Obrist et al. (2011)
B. barbastellus			7.70		Goerlitz et al. (2010)

2.2 Pattern Classification

In this section some pattern classification theories and techniques will be described.

2.2.1 Real-time Tracking and Offline Trajectory Reconstruction

Unlike real-time tracking, this project focuses on offline trajectory reconstruction. Real-time tracking usually utilizes a recursive algorithm which is efficient in computing (Sun et al., 2005) and processes data in a single forward sequential way (Garcia et al., 2007); besides, it has been utilized in many applications such as video surveillance, object searching in video databases, and traffic monitoring and control systems.

The basic concept of real-time tracking is to employ both object features and object dynamics to evaluate object locations in current measurements. The most popular spatial-temporal filter is the Kalman filter. However this is a linear filter which can only model Gaussian noise, and can operate only one task at a time (Kalman, 1960). For coping with above problems, some improvements have been proposed such as the extended Kalman filter, the unscented Kalman filter, and the particle filter (Yang and Zhang, 2007). In recent years, pattern classification has also been utilized in real-time systems. Calderara et al. (2009) proposed a method to detect and model object trajectory by using a multi-camera setup, and to identify either normal and abnormal paths. Bertini et al. (2012) presented a survey for anomaly searching and localization with an unsupervised method using non-parametric modelling.

On the other hand, trajectory reconstruction or smoothing is a batch process which exploits all available data. The aim is to locate interesting objects over recorded data files (Garcia et al., 2007). In the research area of biology, the biologist does not need to know the details of bats' flight characteristics, current positions or a bat group's behaviour in real time. After all, there is little action that the biologist can take even if he or she has this kind of real-time information from bats.

2.2.2 Trajectory Reconstruction using Machine Learning

Similar to the research area of image segmentation which partitions input image into regions considered as homogeneous - and can be implemented by different available techniques such as region-based, edge-based, and cluster based (Jain and Dubes, 1988) - a trajectory reconstruction also can be implemented using different machine learning methodologies.

In the application of an air traffic control (ATC) system, a model based flight trajectory reconstruction was proposed (Garcia et al., 2007). For each flight trajectory, each segment represented a time period when the aeroplane is flying in a specific type of manoeuvre such as uniform, transversal or longitudinal motion, and decelerations. Yang and Zhang (2007) claimed that "the segment says more than the individual hypothesis because higher level components always offer more information on the global structure than lower level primitives." That is why we usually directly break a complete trajectory into evenly distributed segments or extract segmented information indirectly.

A parametric reconstruction filter (IMM tracking filter) for the segmentation that has been described above is shown in figure 4. The filter was constructed with three Kalman filters which worked in parallel and were matched to each manoeuvre. A PART algorithm was selected to extract segmental information for discrimination analysis based on a data mining strategy (Witten and Frank, 2000).

Such model based reconstruction has the advantage of having a less residual, over-fitting effect, and the number of reconstruction samples compared with the conventional interpolation method which using more control points brings more accurate data fitting (Yang and Zhang, 2007). On the other hand, a model based reconstruction is preferred to enable use of specific models with regular motion patterns; however motion paths in some applications are not usually specific and are in fact too complicated to be modelled into regular patterns.

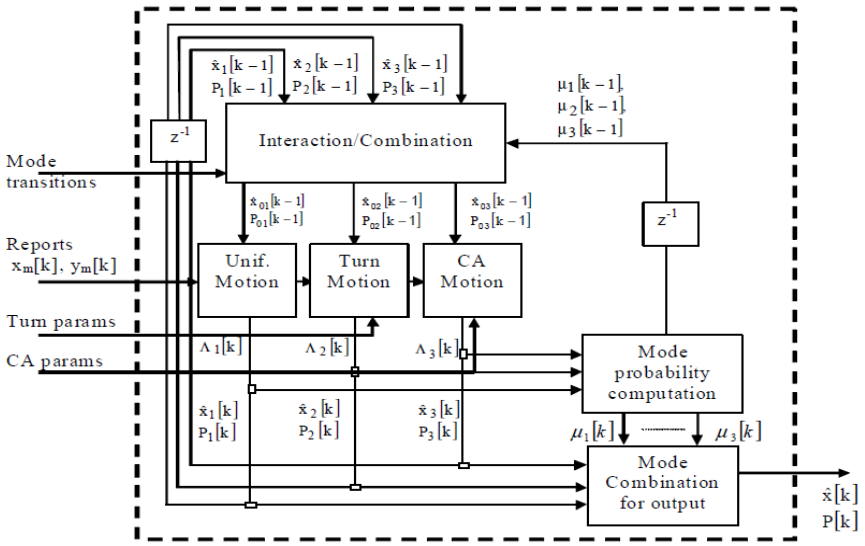


Figure 4. IMM tracking filter for segmentation

Source: Garcia et al., 2007.

In another trajectory reconstruction application, object tracking in sports video scenes, a swimmer's cap tracking was proposed by Yang and Zhang (2007). In the process of clustering, feature extraction was utilized by mean shift seeking (Comaniciu et al, 2000) in the beginning, and followed by an agglomerative hierarchical clustering, utilized to approach the local non-minimum suppression and grouping.

Essentially, objects are grouped or clustered by hierarchical clustering algorithms with sequential partitions; on the other hand, partitional clustering algorithms divide objects into a specified number of groups directly and do not own a clustering structure (Yang and Chen, 2010). Therefore, relying on this clustering structure, hierarchical algorithms are more multi-functional than the latter. For example "a single-link clustering algorithm works well on data sets containing non-isotropic clusters including well-separated, chain-like, and concentric clusters," (Jain et al. ,1999). In contrast, partitional clustering algorithms have the merit of being able to handle considerable data sets because they have a concise structure.

Next, for the dissimilarity measurements, the most popular distance formula for continuous features, the Euclidean distance, was employed in this research where the threshold was set to the projected width of the lane in a swimming pool (Yang and Zhang, 2007), and the formula with two partitions was described as follows:

$$d_2(x_i, x_j) = \left(\sum_{k=1}^{d} (x_{i,k}, x_{j,k})^2 \right)^{1/2}$$

$$= \left\| x_i - x_j \right\|_2$$

where x is a pattern with the feature vector of d, and x_i and x_j are pattern sets. Euclidean distance is useful to check the similarity or dissimilarity of objects in low dimensional space with compact or isolated clusters (Mao and Jain, 1996), while we also need to use weighting or normalization schemes for preventing the incline toward the largest scaled feature (Jain et al., 1999).

Finally, a cubic spline trajectory interpolation was utilized to represent the swimming cap position in occluded situations. The results of the trajectory segmentation and interpolation have been abstracted in figure 5, where the top-left, top-right, and bottom figures represent the hypothesis in the three dimensional space, the extracted trajectory segments, and the final trajectory after interpolation respectively (Yang and Zhang ,2007).

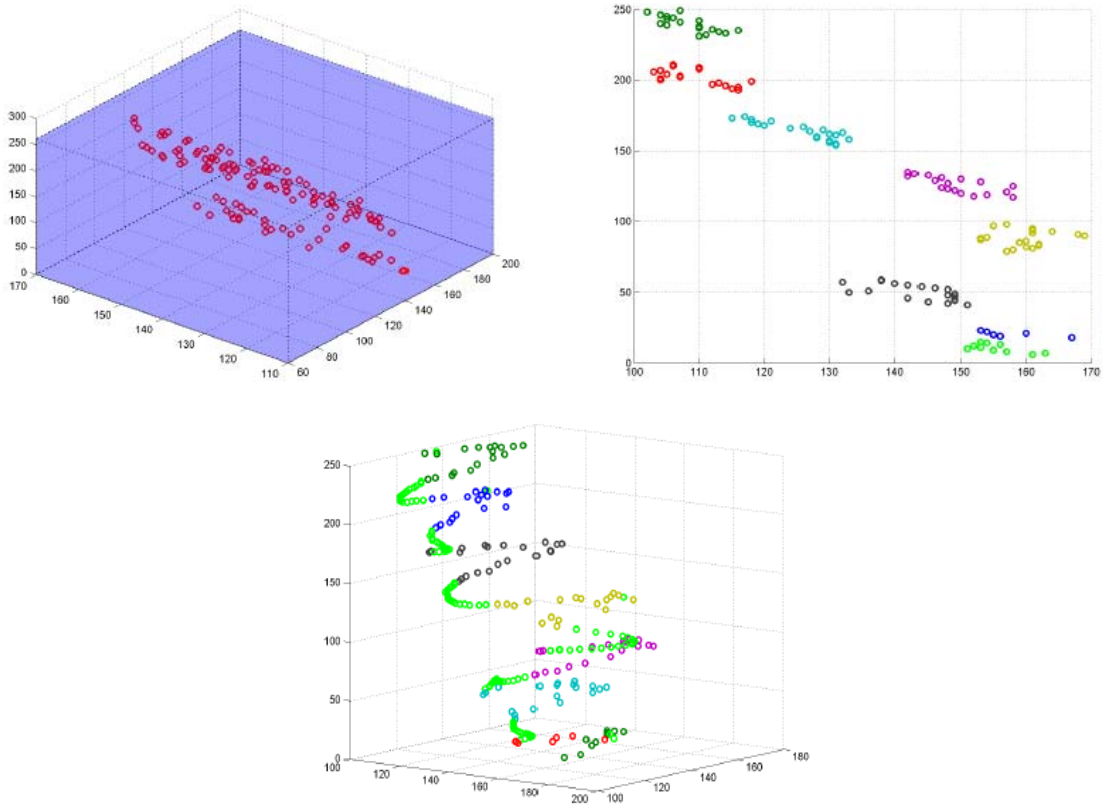


Figure 5. Trajectory segmentation and interpolation where the vertical coordinate and horizontal plane represent frame index and image plane respectively

Source: Yang and Zhang, 2007

Two studies have been described in this section. The former reconstructed the flight trajectory of aeroplanes using a model-based reconstruction method and a data mining classification strategy. Although the degree of freedom of a bat's flight behaviour is much larger than an aeroplane's, a model based filter which utilizes statistical methods can helps us to distinguish which segment set belongs to which bat's trajectory from its noise, echoes and outliers.

On the other hand, the latter study which reconstructs swimming caps reveals a complete guide to using extraction, grouping, and smoothing methods to cope with background noise and long-term occlusions. Admittedly, the research challenges such as background noise and long-term occlusions and objectives are not too largely similar to this study; however this literature gives confidence in utilizing the agglomerative hierarchical clustering method in such low speed requirement, low data volume and high spatial complexity project.

2.3 Smoothing and Interpolation

In this section some smoothing and interpolation methods will be described. Table 4 has revealed some studies using smoothing and interpolating methodologies in their study areas respectively.

Table 4. Smoothing and interpolation methodology utilized in some study areas

Studies	Smoothing methods	Study areas
Erwin et al. (2001)	Linear interpolation	Bat tracking
Garcia et al. (2007)	IMM filter or B-spline interpolation	Aircraft tracking
Garcia et al. (2008)	IMM filter	Aircraft tracking
Songtao et al. (2010)	Cubic spline interpolation	Aircraft tracking
Faria and Dias (2009)	Mean value filter	Gesture tracking
Sun et al. (2005)	B-spline interpolation	Keyframe-based tracking
Tomasi et al. (2003)	Linear interpolation	Gesture tracking
Watts et al. (2001)	Linear interpolation	Bat tracking
Yang and Chen (2010)	Pricewise polynomial interpolation	Pedestrian tracking
Yang and Zhang (2007)	Cubic spline interpolation	Swimming cap tracking

2.3.1 Linear Interpolation

Linear interpolation is the most common mathematical operation in computer graphics (Shirley and Marschner, 2009). If there are two points (x_i, y_i) and (x_{i+1}, y_{i+1}) on a continuous straight line y = P(x), for linear interpolation, each point on this line can be sought easily using the following parametric line equation.

$$P(x) = y_i + \frac{x - x_i}{x_{i+1} - x_i}(y_{i+1} - y_i)$$

However, if two points (x_i, y_i) and (x_{i+1}, y_{i+1}) are separated largely or y = P(x) is not exactly linear, there will be a huge error between the calculated point and real curve when using linear interpolation. In table 4, considering Erwin et al. (2001), Tomasi et al. (2003), and Watts et al. (2001)'s focus on gesture and bat tracking, the motion speed, region, and accuracy of tracking objects are not so high. Therefore, the usage of the linear interpolation method in these applications only for smoothing may be an effective and efficient decision.

2.3.2 Cubic spline Interpolation

In some interpolation applications, the smoothness of the curve is much stressed (e.g. aircraft tracking), and this curve's derivative equation needs smoothness at each corner. Cubic spline can satisfy this requirement, which allows for C^2 continuity that means its position, first and second derivative are continuous (Shirley and Marschner, 2009). If the Cubic spline is used for representing a trajectory, there will be no speed variety between the different trajectory segments; in other words, if the speeds of an object in segments are different, the trajectory will not be continuous. Moreover, another benefit of cubic spline interpolation is that its coefficients bend the spline curve to pass through all data points with no gap nor erratic behaviour as shown in figure 6 (Rogers and Adams, 1990). This characteristic let cubic spline interpolation be suitable for low noise applications.

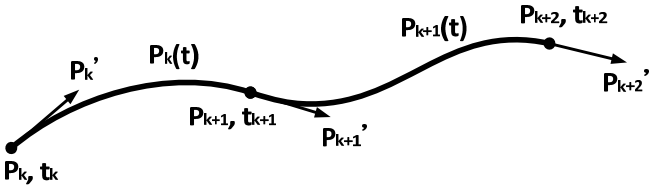


Figure 6. Two piecewise cubic spline segment curve

Furthermore, its equation has been shown as follows (Rogers and Adams, 1990),

$$P_k(t) = \sum_{i=1}^{4} B_{k,i} t^{i-1} \quad , t_1 \le t \le t_2$$

where $P_k(t)$ represents the position on the curve of cubic spline interpolation segment k; t1 and t2 are parameter values at the beginning and end positions of segment k respectively; $B_{k,i}$ are constant coefficients determined by the four parameters described as follows:

$$\begin{cases} B_{k,1} = P_k \\ B_{k,2} = P'_k \\ B_{k,3} = \dfrac{3(P_{k+1} - P_k)}{t^2_{k+1}} - \dfrac{2P'_k}{t_{k+1}} - \dfrac{2P'_{k+1}}{t_{k+1}} \\ B_{k,4} = \dfrac{2(P_k - P_{k+1})}{t^3_{k+1}} + \dfrac{P'_k}{t^2_{k+1}} + \dfrac{P'_{k+1}}{t^2_{k+1}} \end{cases}$$

Cubic spline interpolation is favourable in computer graphics and the following reasons are given: minimum-curvature interpolation (it presents the smoothest curve with minimum

curvature because of its low order); good symmetry (both positions and derivative values at start and stop points can be specified), and a good trade-off between smoothness and computational costs (Shirley and Marschner, 2009).

2.3.3 B-spline Interpolation

For more complicated curve shapes, in practice, the traditional interpolation method may not be a suitable solution because they have poor parametric continuity and no control point. On the other hand, control points in approximating curves such as Bezier and B-spline can shape their curves directly and bring about a better curve behaviour and local control; nevertheless approximated curves have difficulty in passing through these control points (Shirley and Marschner, 2009). Therefore, in some applications, if tabulated values need to be passed through by a trajectory, approximating schemes may not be a wise choice. Moreover, unlike a Cubic spline curve defined by positions and derivate values at the beginning and stop points, a B-spline curve is defined by control points and basic function, and its definition has been shown in the following equation (Shirley and Marschner, 2009):

$$P(t) = \sum_{i=1}^{n+1} B_i N_{i,k}(t) \quad , t_{min} \leq t < t_{max} \quad 2 \leq k \leq n+1$$

where B_i are control points (position vector), N_i are the normalized basis function, and n are the number of control points.

In summary, there are three popular interpolation methods which have been described in this section: linear interpolation, cubic spline interpolation and B-spline interpolation. For the application of smoothing, interpolation is usually the last task or block in the system followed by visualization for human inspection. In this project, smoothing for human visualization will be implemented by linear interpolation because the curvature in bat's flight path is not so complicated and there are not too many missing points in a typical flight path. With some species like *Nyctalus leisleri*, their flight paths can even be seen as a straight line; besides, we know that a bat's flight speed is not that fast when compared with the speed of ultrasound.

However, the application of interpolation that really concerns this study is not smoothing. Interpolation also can be utilized in the learning cycle of clustering. In a cluster, interpolation can help to approach a more accurate evaluation of the segment orientation vectors at both beginning and stop points. Furthermore, higher order interpolation can help to re-sample

segments for improving the performance of features for extraction and modelling. If the utilization of the interpolation is in the learning cycle of clustering, not just smoothing for human inspection, then requirements of smoothness and accuracy will become much more important; therefore, the spline interpolation method may take on an important role in this project. Furthermore, if we consider that all data points need to be passed through by reconstructed curves, cubic spline interpolation method will be the optimum choice for this study.

3. Methods

In this chapter some important units, methodologies and assessments will be described.

3.1 Input data analysis and pre-processing

Bat's flight trajectories, including sequential time and three dimensional flight position information are recorded by two single sensor arrays of four microphones respectively which are spaced approximately from 5 to 10 meters apart as shown in figure 7.

Figure 7. Multi sensor arrays in Clifton

Source: Picture is provided by Dr. Holger R. Goerlitz

These sequential timestamps, recorded in good quality by both sensor arrays, are calculated and synthesized into one ORT file which lists sequential timestamps with higher accuracy than those recorded by individual arrays listed in a WNK file. However, sequential timestamps listed in an ORT file are not always enough; for example, if one bat changes its trajectory, flies on the side of one of the sensor arrays, will bring a large difference of two distances between the bat and two sensor arrays. These unbalanced signal strengths from the two sensor arrays cause some important spatial data points to disappear in the ORT file. In this kind of situation, WNK files will be used to compensate for the multi sensor arrays' calculation error in the pre-processing part of our algorithm. Moreover, temporal resolution deviation and physical reliability from biological knowledge will also be corrected and considered in the pre-processing.

Pre-processing is described by figure 8 which includes a coordinate system converter, and a temporal and spatial data points rectifier. The former converts the raw data in the WNK file represented by spherical polar coordinates to Cartesian coordinates, and the latter implements two channel unbalance checking, temporal resolution checking, sorting and monotone constraining, and physical reliability checking.

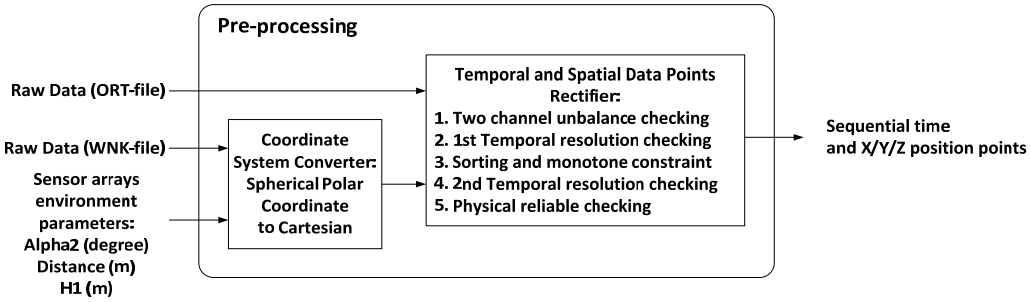


Figure 8. Pre-processing block

In the coordinate system converter, the environment parameters of the placement of the two sensor arrays (A1 and A2) including horizontal orientation degrees (Alpha1 and Alpha2), distance between two arrays (Distance), and the height of central microphones (H1 and H2) described in figure 9 and figure 10 are utilized to construct their spherical polar coordinates, as shown in figure 11.

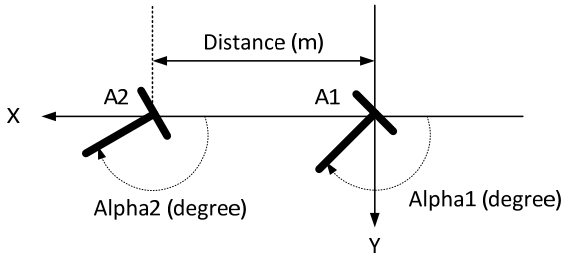


Figure 9. Placement of multi sensor arrays in top view

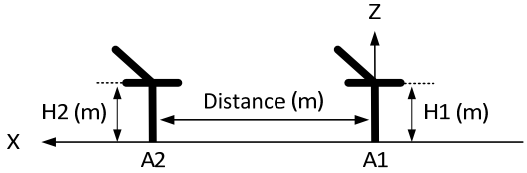


Figure 10. Placement of multi sensor arrays in side view

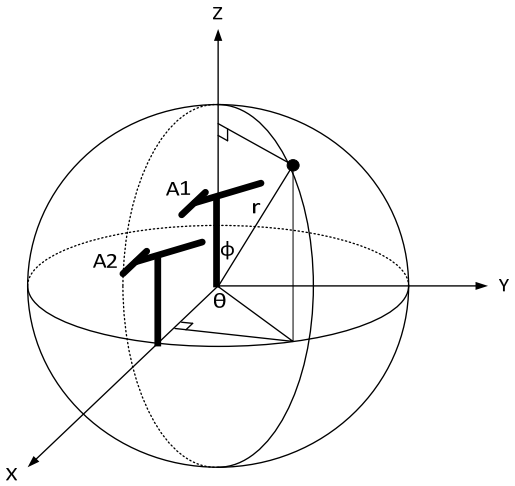


Figure 11. Multi sensor arrays and spherical polar coordinate

The Cartesian coordinates (X, Y, Z) are related to the spherical polar coordinates (r, Θ, Φ) by the following equations:

$$X = r \cdot \cos\Theta \cdot \sin\Phi$$
$$Y = r \cdot \sin\Theta \cdot \sin\Phi$$
$$Z = r \cdot \cos\Phi$$

Thus, raw data both from WNK and ORT files can be processed in the same coordinate system. During each computational process of the trajectories' reconstruction, two WNK files which recorded temporal and spatial data points from two sensor arrays and one ORT file which recorded temporal and spatial data points synthesized from two sensor arrays will be processed. Examples of different formats of the raw data file are extracted and shown from table 5 to table 7.

Table 5. ORT file format and raw data example

T1[s]	T2[s]	X[m]	Y[m]	Z[m]
3.2609	3.2540	8.49	21.01	5.62
3.4406	3.4358	8.20	20.51	5.94

Table 6. WNK file format and raw data example from sensor array A1

T1[s]	Phi	Theta	r[m]
3.2609	-3.5	11.3	6.58
3.3580	-22.8	51.1	2.05

Table 7. WNK file format and raw data example from sensor array A2

T2[s]	Phi	Theta	r[m]
3.2540	-11.6	9.4	18.49
3.4358	-10.8	11.0	35.42

In the temporal and spatial data points rectifier, raw data from ORT and WNK files will be merged in the beginning to compensate for the unbalanced signal strengths received from the two sensor arrays. Before merging, a simple temporal resolution check will be implemented to delete the duplicate data points between the ORT and WNK files, and further to restrict the number of data points from the WNK files. Then, these temporal and spatial data points will be sorted for monotonic timing sequence. Besides, the second temporal resolution checking will be implemented again. If the timing difference of neighbouring temporal points is too small, for example less than 0.1ms, a reasonable timing shift will be inserted between these two neighbouring temporal points for a more efficient and reliable computation in the latter phase. Finally, physical reliability from biological knowledge will also be checked. For example, the click position of height in Z coordinate must be larger than zero.

Finally, the pseudo code of this algorithm is described in the following.

Table 8. Pseudo code of pre-processing

- Load ORT file and WNK file
- Check empty
- Close ORT file and WNK file
- For i = 1 to length of WNK file
  - Retain WNK(i) if the timing difference between ORT(:) and WNK(i) > 100ms
- End For
- Load environment parameters of placement
- Convert WNK file's coordinate system from spherical polar to Cartesian
- ORT&WNK file = merge and sort ORT file and WNK file
- For i = 1 to length of ORT&WNK file -1
  - Constrain timing difference between ORT&WNK(i) and ORT&WNK(i+1) > 0.1ms
  - Constrain Z > 0
- End For

3.2 Interpolation

The bat's flight trajectory is represented by bat clicks; from the point of view of sampling theory, these bat clicks are sampling points, and the continuous flight curve is sampled by the bat's vocal system. Unlike general sampling systems, this biological sampling system is hard to predict and model; for example, the input frequency and sampling bandwidth of the quantizer, the bat's vocal system, differ between each bat at each time – there are too many degrees of freedom. If we cannot model this sampling system precisely, it is hard to transform these discrete data points to a continuous trajectory correctly by using traditional sampling theory and methodology.

Moreover, our microphone arrays system is another sampling system which receives many acoustic sampled signals from the same bat, different bats, echoes and noise in a short period of time. All sampled data points are mixed together and are hard to distinguish and analyze. What we can do is use some graphics skills to simulate and reconstruct the original flight curves, and discover those most probable trajectories which fit the biological features best. Therefore, sampling theory and complicated biological modes can be abandoned in this project; in contrast, interpolation skills will be utilized in this work to transfer discrete data points to a continuous trajectory. Additionally, the bats' flight features and models will be extracted and constructed from prior work - just a simple statistical modelling process - and will be described in next section in detail.

Information and features could be extracted from segments much more readily than from discrete data points. However, segments established by linear interpolation are not sufficient in this work, because the trajectory of a bat's flight is always continuous and smooths thoroughly, even on tortuosity. There must be a large deviation between the prediction and the real case if a linear interpolation segment is considered here. For example, three bat click points are recorded by our system, and their time and X/Y/Z data points are P1={4.44, 7.76, 6.54, 7.01}, P2={4.73, 8.53, 0.68, 2.67} and P3={5.14, 6.76, -1.81, 6.80}. Their curves are reconstructed by using both linear and higher degree polynomial interpolations shown in figures 12 and 13.

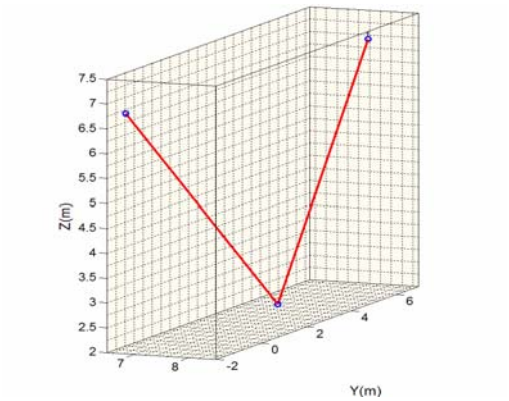


Figure 12. Linear interpolation

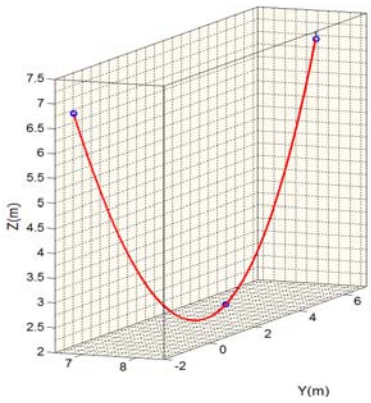


Figure 13. Higher order polynomial interpolation

The comparison of features such as path length, maximum and minimum speeds, and maximum curvature between the different interpolations is described in following table. It can be concluded the reliability of features such as path length, minimum speed and maximum curvature from polynomial interpolation is much better than that from linear interpolation.

Table 9. Comparison between two interpolation methods

Interpolation	Path Length (m)	Max Speed (m/s)	Min Speed (m/s)	Max Curvature (1/m)
Figure 12.	12.47	25.51	12.36	0.10
Figure 13.	13.74	25.69	15.31	0.11
Error (%)	10.23	0.72	23.82	9.80

Cubic spline interpolation is based on the third degree polynomial interpolation, and which cubic spline interpolation can meet the requirement of allowance for C^2 continuity has been described before; besides, the most important point is that its coefficients bend the curve to pass through all data points with no gap nor erratic behaviour. It is hoped that the reconstructed trajectory can pass through its own data points similar to the biologist's manual calculation. Of course, there will be some errors between the real click positions and the temporal and spatial positions recorded by the microphone sensor arrays, but these little deviations may not be a major problem. Furthermore, in terms of cost consideration, the cubic spline interpolation is the minimum-curvature interpolation, and has a good trade-off between smoothness and computational costs (Shirley and Marschner, 2009). On the other hand, if we consider a higher order (larger than 3) polynomial approximation using the least square method (Matlab function: polyFit) to guarantee all possible data points are passed through, we need to pay more computation and stability costs.

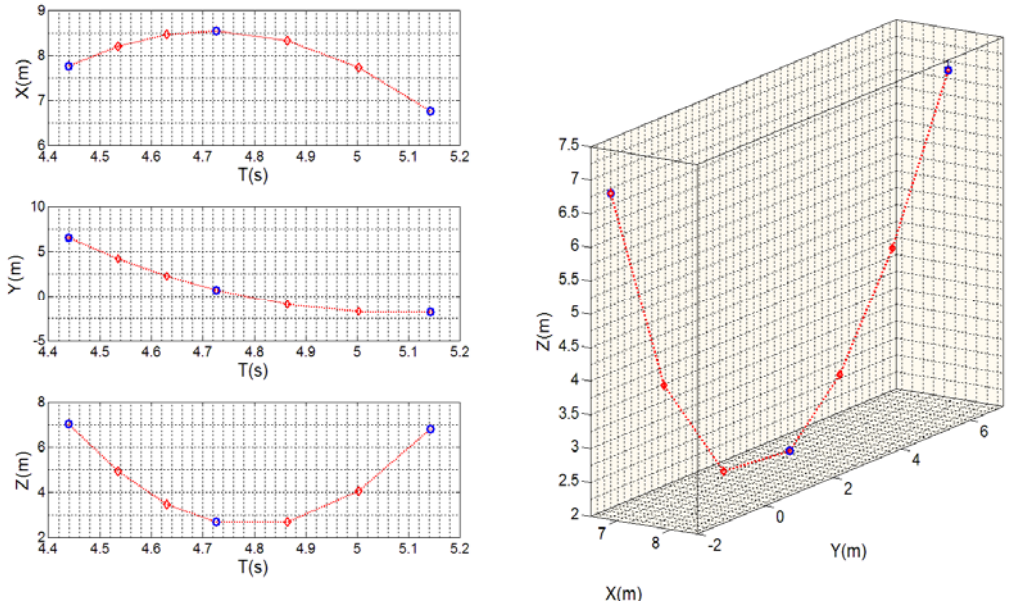


Figure 14. The implementation of cubic spline interpolation in 3D space

3D cubic spline interpolation in Matlab cannot be implemented directly because the relationship between interpolated points and arguments of the spline function in Matlab should be monotonic and uniquely specified. Therefore, segments projected to the X/Y/Z coordinates are interpolated on the time domain firstly and then combined later, as shown in figure 14, where the number of interpolation points in each segment is set to 2.

The choice of the number of the interpolation points is important. Choosing larger re-sample points brings a better accuracy, but takes a huge computational time; on the other hand, choosing less points saves more computational costs but may bring poor results of feature extraction. Of course, we can use some iteration algorithms to find out a local best setting, but it also takes lots of time; especially, it may be a disaster if the raw data is large. In this first version of this project, the number of interpolation points is fixed to 50. This constant value guarantees the reliability to a certain degree for most segments in the trajectories; however, it also takes a large computational time for most segments. On the other hand, for some trajectories whose subset segments have a large unbalanced path length compared with their neighbours, 50 points may not be enough to guarantee their captured features are reliable. Optimization of computational time and improvement of reliability will be discussed in the next chapter.

Table 10 shows the pseudo code of interpolation where N is 50 in this chapter.

Table 10. Pseudo code of interpolation

- Load time and X/Y/Z points : T, X, Y and Z
- Load the number of interpolation points: N points
- For i = 1 to length of time points -1
  - Split the time points' interval(i) into N partitions equally: N*dt(i)
- End For
- Interpolated X points = spline (T, X, dt);
- Interpolated Y points = spline (T, Y, dt);
- Interpolated Z points = spline (T, Z, dt);

3.3 Prior Flight Model

To reconstruct a bat's trajectory automatically, our algorithm needs to recognize data points or segments which belong to bat clicks from echoes and outliers. Therefore, capturing bat features from prior results and then constructing a bat flight model are important steps in this project. In the literature review, there are some flight parameters of different species collected, such as in Table 3; however, they are not complete and adequate enough to distinguish multiple and more complicated trajectories. Therefore, hundreds of bat's raw data files (FLG files) from Dr. Holger R. Goerlitz are arranged and analyzed in this project to build a more complete and accurate *Nyctalus leisleri* bat model.

Ten features captured and extracted from data points in each FLG file are labelled from x1 to x10 and described in the following table. Each FLG file includes one to several hand-reconstructed bat's trajectories, of which data points are recorded over a short period of time and processed by biologists.

Table 11. Bat's flight features

x1 = maximum flight speed.
x2 = minimum flight speed.
x3 = average flight speed in x coordinate.
x4 = average flight speed in y coordinate.
x5 = average flight speed in z coordinate.
x6 = maximum flight curvature.
x7 = maximum click to click distance.
x8 = minimum click to click distance.
x9 = maximum click frequency.
x10 = minimum click frequency.

For capturing the above ten features, point-to-point distance, point-to-point velocity, three points curvature radius and point-to-point frequency are calculated; specifically, before calculating these parameters, each prior trajectory is re-sampled (interpolated by cubic spline interpolation) for better accuracy. Furthermore, unlike the other parameters, the calculation of the 3D curvature radius is much more complex. Curvature means how fast a curve changes its direction, and also represents the rate of change of the direction of a curve at one point to another point with respect to the distance between these two points along the curve. If our curve y = f(x) is on a two coordinates plane, then the curvature (K) represents to $d\vartheta/dc$, where ϑ is the tendency of the captured tangent, and c is the arc length of two points on the curve (Kline, 1998). The radius (R) of the curvature of a curve at one point is defined to be 1/K, as shown in figure 15.

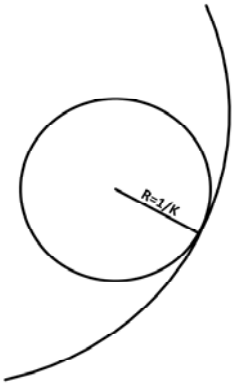


Figure 15. Radius of curvature

Therefore, the formula of the radius of curvature is often represented by the following (Kline, 1998):

$$R = \frac{1}{K} = \frac{dc}{d\vartheta} = \frac{(1 + y'^2)^{3/2}}{y''}$$

Unfortunately, this formula cannot be utilized in 3D space directly because curvature or radius of curvature cannot be combined or captured from its body double projected to X-Y, X-Z, and Y-Z planes. Additionally, a bat's trajectory in 3D space is not always monotonic so this differential equation is difficult to be implemented here. In contrast, in this project, a 3D circle which passes through three adjacent points on the curve (concyclic points) is constructed, and then the radius of curvature can be captured by solving simultaneous equations with three degrees and four unknowns as following, where O is centre of the circle, P1, P2 and P3 are adjacent concyclic points, and R1, R2 and R3 are radiuses respect to each

point:

$$\begin{cases}(O_x - P1_x)^2 + (O_y - P1_y)^2 + (O_z - P1_z)^2 = R1^2 \\ (O_x - P2_x)^2 + (O_y - P2_y)^2 + (O_z - P2_z)^2 = R2^2 \\ (O_x - P3_x)^2 + (O_y - P3_y)^2 + (O_z - P3_z)^2 = R3^2 \end{cases}$$

Ideally if *dc* (minimum length of arc length on curve) is close to 0, the centre of the circle is unique, and R1, R2 and R3 are equal. However, in our case, the number of interpolation points cannot be infinitely great, and there may be no solution. So, Moore-Penrose's "pseudoinverse" is utilized here to compute a best fitting- least squares solution.

The features' mean and standard deviation are extracted from a hundred trajectories for constructing Gaussian models as described in figure 16, where μ denotes mean, the standard deviation (σ) is the root of its variance, σ^2, and X means the featured variable. To obtain a biological model (B), outliers are filtered out from measurement model (M). That means some particularly noisy data points are removed before the statistical calculation.

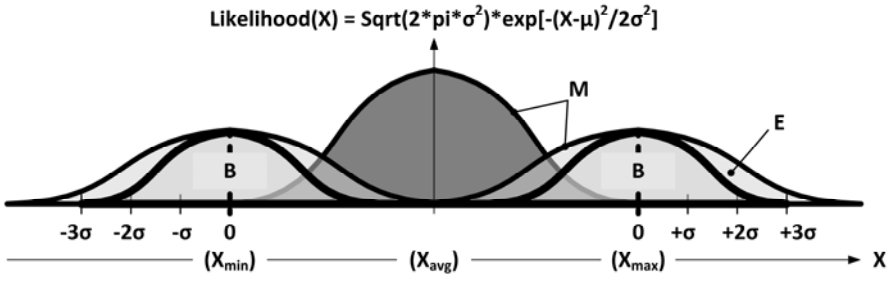
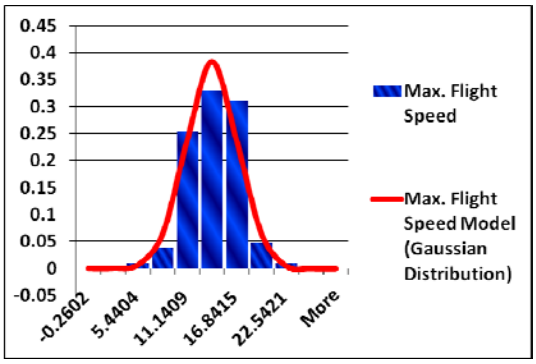
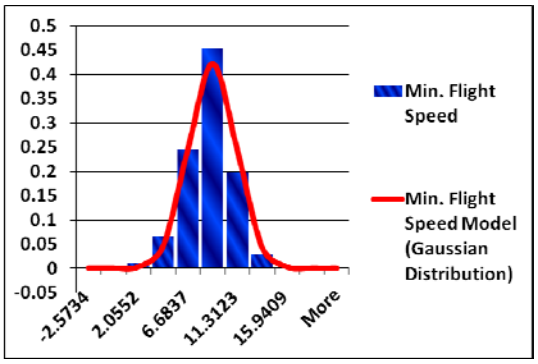


Figure 16. Gaussian model

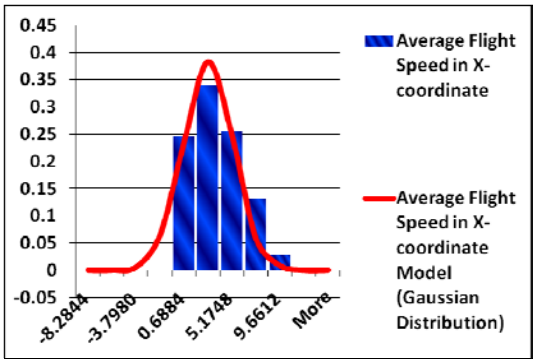
Figure 17 shows both normalized accumulated distributions from FLG files (blue bar) and prior bat flight's Gaussian models (red line) over ten features. In this project, we utilize Gaussian distribution to model each flight feature; however, the real situation is that not all the flight features' accumulated distributions are similar to Gaussian distribution such as that shown in figure 17(f), maximum flight curvature. In other words, there will be some difficulties when measuring the likelihoods of maximum flight curvature of a few reconstructed cases.


0.45
0.4
0.35
0.3
0.25
0.2
0.15
0.1
0.05
0
-0.05
-0.2602
5.4404
11.1409
16.8415
22.5421
More
Max. Flight Speed
Max. Flight Speed Model (Gaussian Distribution)

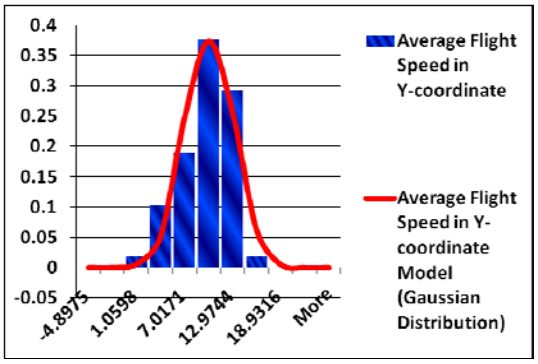
(a)


0.5
0.45
0.4
0.35
0.3
0.25
0.2
0.15
0.1
0.05
0
-0.05
-2.5734
2.0552
6.6837
11.3123
15.9409
More
Min. Flight Speed
Min. Flight Speed Model (Gaussian Distribution)

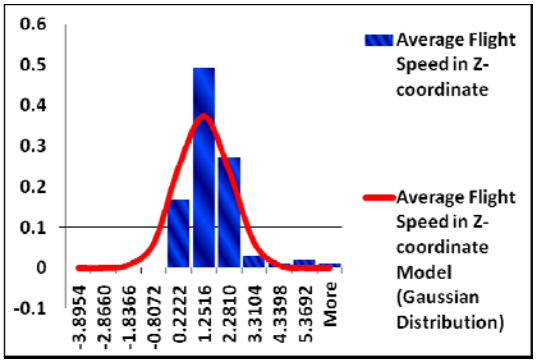
(b)


0.45
0.4
0.35
0.3
0.25
0.2
0.15
0.1
0.05
0
-0.05
-8.2844
-3.7980
0.6884
5.1748
9.6612
More
Average Flight Speed in X-coordinate
Average Flight Speed in X-coordinate Model (Gaussian Distribution)

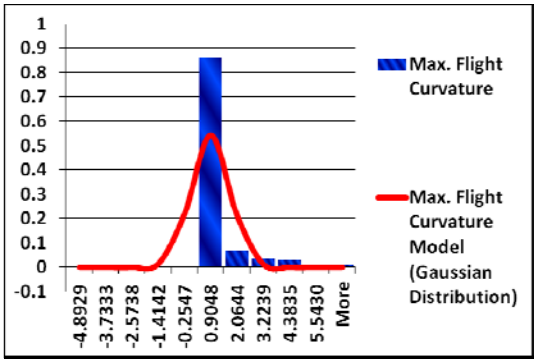
(c)


0.4
0.35
0.3
0.25
0.2
0.15
0.1
0.05
0
-0.05
-4.8975
1.0598
7.0171
12.9744
18.9316
More
Average Flight Speed in Y-coordinate
Average Flight Speed in Y-coordinate Model (Gaussian Distribution)

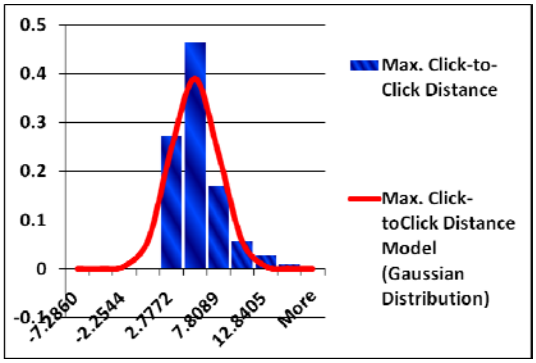
(d)


0.6
0.5
0.4
0.3
0.2
0.1
0
-0.1
-3.8954
-2.8660
-1.8366
-0.8072
0.2222
1.2516
2.2810
3.3104
4.3398
5.3692
More
Average Flight Speed in Z-coordinate
Average Flight Speed in Z-coordinate Model (Gaussian Distribution)

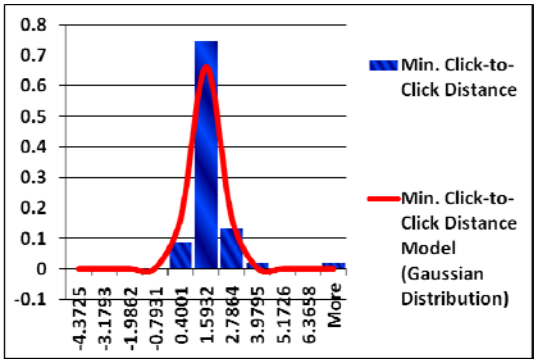
(e)


1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
-0.1
-4.8929
-3.7333
-2.5738
-1.4142
-0.2547
0.9048
2.0644
3.2239
4.3835
5.5430
More
Max. Flight Curvature
Max. Flight Curvature Model (Gaussian Distribution)

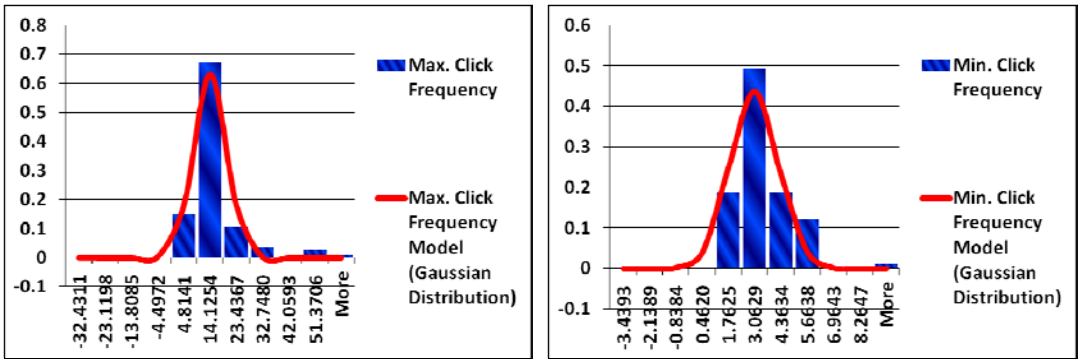
(f)


0.5
0.4
0.3
0.2
0.1
0
-0.1
-2.2860
-2.2544
2.7772
7.8089
12.8405
More
Max. Click-to-Click Distance
Max. Click-toClick Distance Model (Gaussian Distribution)

(g)


0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
-0.1
-4.3725
-3.1793
-1.9862
-0.7931
0.4001
1.5932
2.7864
3.9795
5.1726
6.3658
More
Min. Click-to-Click Distance
Min. Click-to-Click Distance Model (Gaussian Distribution)

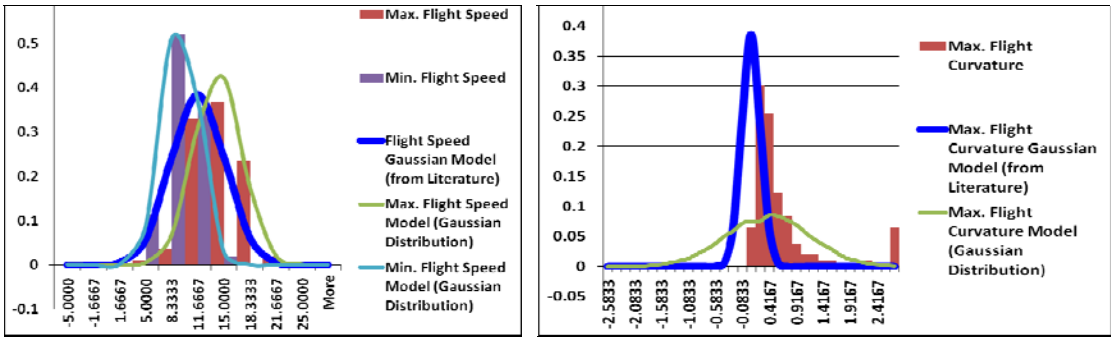
(h)



(i) (j)

Figure 17. Normalized accumulated distributions and prior bat flight models. (a)-(b) max. and min. flight speeds; (c)-(e) average flight speeds; (f) max. flight curvature; (g)-(h) max. and min. click-to-click distances; (i)-(j) max. and min. click frequencies

Figure 18 (a) compares both normalized accumulated distributions (brown and purple bars) from FLG files and prior bat flights' Gaussian models (thin green and blue lines) on maximum and minimum flight speed with bat behaviour models (thick blue line) which refer to previous literature. The speed accumulated distributions and Gaussian models are similar to the previous literature. By contrast, figure 18 (b) compares both normalized accumulated distributions (brown bar) from FLG files and prior bat flights' Gaussian model (thin green line) on maximum flight curvature with bat's behaviour model (thick blue line) which refers to previous literature. It reveals the accumulated distribution of bat flight curvature matches the literatures', and that the curvature measurement methodology is reliable. Nevertheless, our bat flight Gaussian model is imperfect, and there may be issues in the reconstruction of a few cases.



(a) (b)

Figure 18. Comparison of accumulated distribution, prior bat flight models and literature sssbehaviour models on features of max. and min. flight speeds and max. curvature

Finally, their correlation is analyzed in table 12, and the clustered result of bat flight feature indicators is shown in figure 19. If indicators have a higher positive or negative correlation

and are orthogonal with each other, R clustering can be implemented and help to reduce the complexity. This result shows there are only two indicators that have a higher correlation (maximum click-to-click distance and minimum click frequency have a larger correlation), and the benefit of reduction is not large. On the other hand, this result demonstrates these indicators have great independent representations.

Table 12. Correlation coefficients

	x1	x2	x3	x4	x5	x6	x7	x8	x9	x10
x1	1									
x2	0.190	1								
x3	-0.206	0.255	1							
x4	0.474	0.436	-0.405	1						
x5	-0.208	0.005	0.241	-0.308	1					
x6	0.211	-0.402	-0.128	-0.161	0.011	1				
x7	0.205	0.274	0.060	0.000	-0.020	-0.079	1			
x8	-0.212	0.323	0.180	0.030	0.253	-0.233	0.361	1		
x9	0.410	-0.311	-0.247	-0.134	-0.035	0.553	-0.158	-0.443	1	
x10	-0.076	0.101	0.146	0.054	0.077	-0.020	-0.718	-0.271	0.037	1

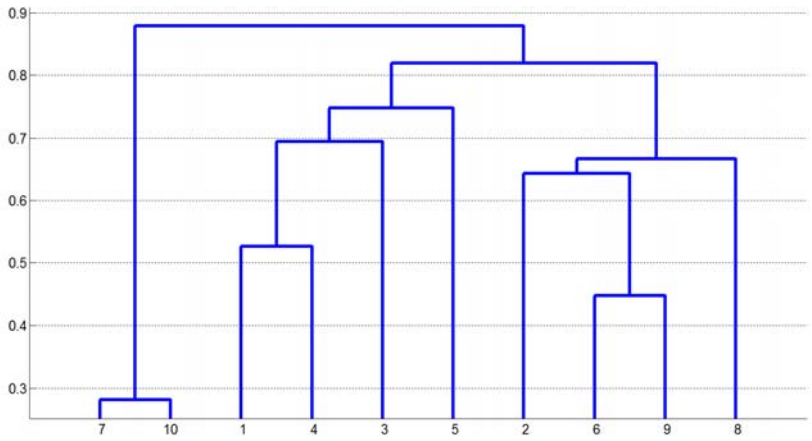


Figure 19. Clustered result of bat flight feature indicators

3.4 Trajectory's Segment Subset Selection Algorithm

A trajectory's segment subset selection algorithm based on clustering and branch-and-bound methods is implemented to select the best subset of segments from the trajectory's segment set. The difficulty of segment subset selection is in the selection of a subset of segments with the best combination and the optimum bat behaviours from a large set of segments. Brute force evaluation of all the subsets is unfeasible. In contrast, the branch-and-bound algorithm is efficient. It rejects trajectories with lower probability without interviewing all the subsets, and guarantees that the selected trajectory is the globally best representation (Narendra and Fukunaga, 1977). Furthermore, the model based branch-and-bound helps us select trajectories which match bat flight models; however, it is hard to distinguish trajectory clusters. In contrast, agglomerative clustering using an unweighted Jaccard distance, and longest path selection method are utilized in this project to help us select the best trajectory in each cluster, as shown in Figure 20.

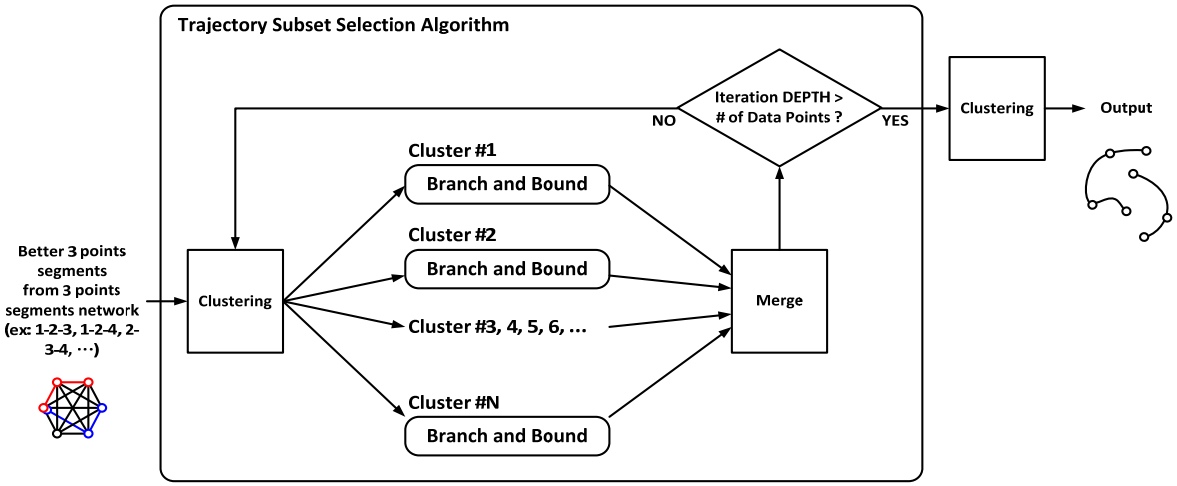


Figure 20. System view of trajectory's segment subset selection algorithm

In figure 20, input patterns (the better 3 point segments) are processed by a 3 point segment selection algorithm which will be described below. The trajectory's segment subset selection recursion starts at the first clustering block and ends at the decision making block which compares the iteration depth with the number of raw data points. If the iteration depth is larger than the number of raw data points, recursion will be terminated, and segments will be clustered by the second clustering block and then passed to the next stage. In each iteration, segments are clustered firstly, and then these clusters are processed by branch-and-bound algorithms individually. Finally, these extended and pruned segments are merged and passed to the next stage, either the first clustering or the second clustering.

3.4.1 3 Point Segment Selection Algorithm

Before the trajectory's segment subset selection, sequential time and X/Y/Z position points are filtered by a 3 point segment selection algorithm, as shown in figure 21.

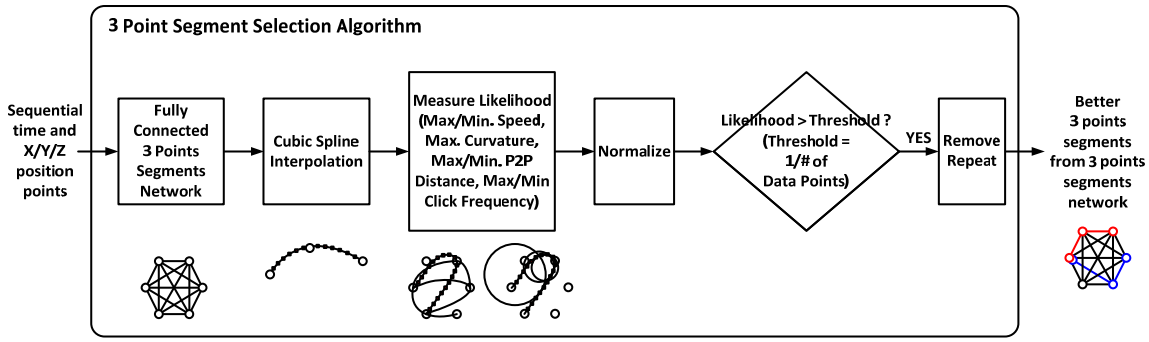


Figure 21. 3 point segment selection algorithm.

In this process, data points are fully connected to the 3 point segments network and interpolated in the beginning. Then, the bat flight parameters are calculated and substituted into the bat flight Gaussian models to get the likelihood of each flight feature. A maximum likelihood method is utilized here to obtain a representation of the whole features. For a more efficient computation, a logarithm is utilized to replace the production operation which is described in the following equation, where i is from 1 to 7, and L(i) represents max. flight speed, min. flight speed, max. flight curvature, max. click-to-click distance, min. click-to-click distance, max. click frequency and min. click frequency respectively.

$$L = \sum_{i=1}^{7} \log L(i)$$

Finally, the likelihoods of all segments are normalized and compared, and then some segments with lower likelihoods or repeated segments are removed.

3.4.2 Branch-and-bound Algorithm

In a branch-and-bound algorithm, as shown figure 22, segment patterns are branched sequentially and thoroughly from their current nodes to the end node of the raw data points. Then, a cubic spline interpolation and maximum likelihood method are implemented, similar to the utilization in the 3 point segment selection algorithm. The threshold in making the decision block here is defined as the maximum 1% of whole segments. Furthermore, if segments are smaller than the threshold, these extended segments will be pruned and become the original segment patterns in the beginning. On the other hand, if the segments are larger than the threshold, these extended segments will be kept and its repeat will be checked together with other pruned segments.

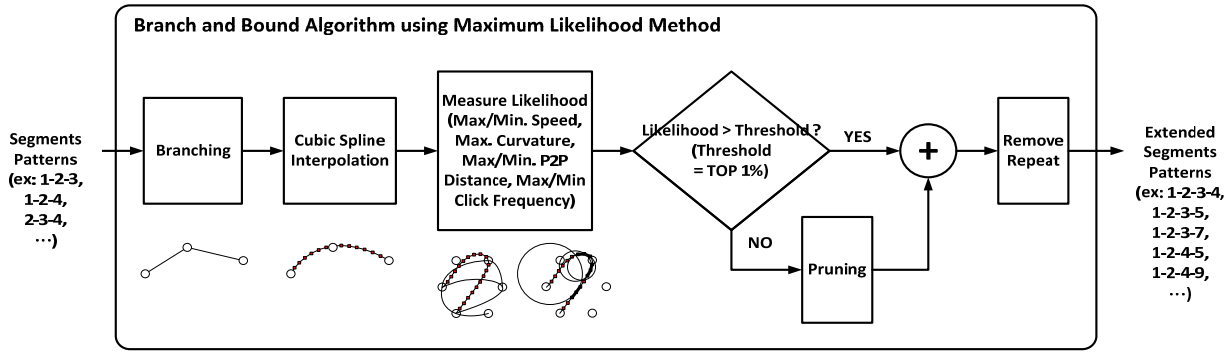


Figure 22. Branch-and-bound algorithm

The pseudo code of branch-and-bound algorithm is described in table 13. There is one branch-and-bound example shown in figure 23, where the initial 3 point segment is 1-2-3, and the length of the raw data points is from 1 to 7. In this example, after the three times iterations was operated, the longest segments pattern is 1-2-3-4-6. Furthermore, we can find that the pruned segment patterns are kept for branching in the next term, such as the 1-2-3 in iteration 1, and both the 1-2-3 and 1-2-3-4 in iteration 2. After branching, the likelihood of each extended or pruned segments pattern will be re-calculated, and the rank of all patterns will be re-sorted; therefore, the branched segments pattern this time may have a lower likelihood the next time. By contrast, the pruned segments pattern this time may be revived the next time, such as the 1-2-3-5 in iteration 2.

Table 13. Pseudo code of branch-and-bound

- Set default span value = 1
- For i = 1 to the number of input segments patterns
  - Current branched segments patterns (span) = input segments patterns (i)
  - If the value of the end node of current branched segments patterns (span) == the value of the end node of raw data points
    - Cubic spline interpolation
    - Measure likelihood
    - span = span + 1
  - Else
    - For j = the value of the end node of current branched segments patterns(span)+1 to the value of the end node of raw data points
      - The next node of current branched segments patterns (span) = j
      - Cubic spline interpolation
      - Measure likelihood
      - span = span + 1
    - END For
  - END If
- End For
- Threshold = the best 1% of the normalized likelihood of all branched segments patterns
- For i = 1 to the number of branched segments patterns
  - If the likelihood of branched segments patterns (i) < threshold
    - the value of Maximum (branched segments patterns (i)) = 0
  - END If
- END For
- Check repeat

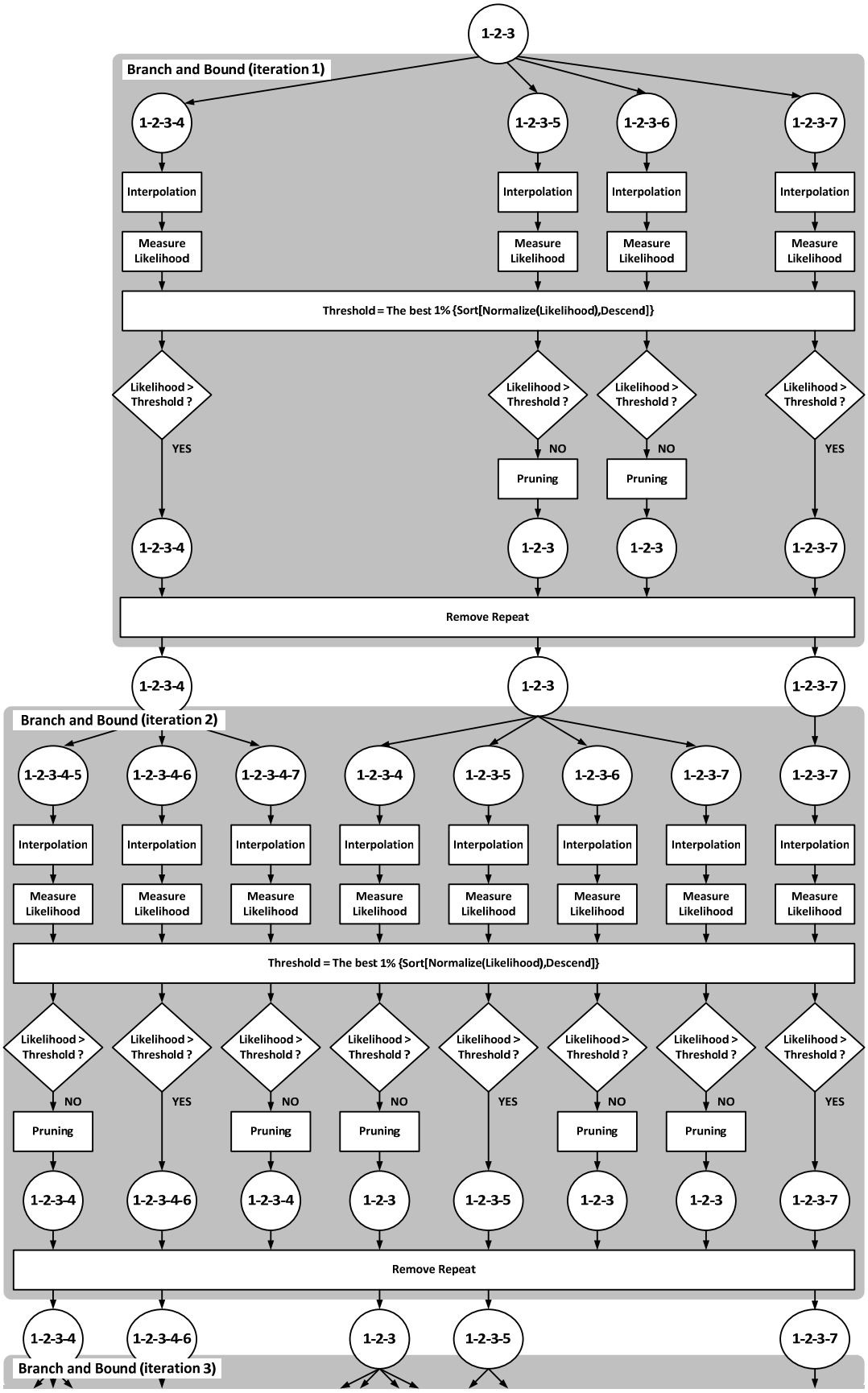


Figure 23. Branch-and-bound example

3.5 Clustering

A model based branch-and-bound algorithm helps to select and construct the best trajectory sets without exhaustive search; however, it cannot recognize which trajectory belongs to the same bat and the number of trajectory clusters. Bifurcation checking is a method that can help to define the relationship between trajectories – i.e. identify which trajectory belongs to which trajectory cluster.

Considering a parameter-dependent nonlinear system utilized in our application, the equation of the general form is described in the following:

$$f(w, \lambda) = 0$$

where $w \in R^3$ represents to X, Y and Z coordinates, and $\lambda \in R^1$ represents the physical factor -

time, $f : R^{3+1} \rightarrow R^3$ and f is sufficiently smooth (Kanzawa and Oishi, 1999). If the number of solutions (w, λ) of $f(w, \lambda) = 0$ varied with respect to λ, we called this function $f(w, \lambda)$ has the phenomenon of bifurcation, and the point (w_0, λ_0) is a bifurcation point or singular point as points near $f(w_0, \lambda_0)$ have bifurcation phenomena. The rigorous definition of the bifurcation point is that any neighbour of the point (w_0, λ_0) on the path of solutions of $f(w_0, \lambda_0) = 0$ is a solution but not lying on the path of solutions of $f(w_0, \lambda_0) = 0$ (Crandall, 1971). For calculating the bifurcation point in this project, we simplify the numerical computation of nonlinear equations of each trajectory by measuring the relationship of the trajectories' intersect.

If two bats' trajectories share the same bifurcation point, we can conclude that they are in the same trajectory cluster as shown in figure 24 where point 9 and 16 (red circles) are bifurcation points of Bat 1 and Bat 1'; additionally, the trajectories of Bat 1 and Bat 1' are classified into the same trajectory cluster. Therefore, the distance between trajectory clusters will be derived from the number of bifurcation points. The representation of click position points of a trajectory is a series of 1's and 0's which relates to a list of raw data points' indexes in chronological order, and can be defined as binary variables. If we want to measure the number of bifurcation points, the similarity of binary variables can be calculated, and the Jaccard similarity coefficient is the most widely used method (Cheetham and Hazel, 1969). Jaccard similarity coefficient may be expressed in several ways, and a common approach is to write it as follows (Urbani, 1980):

$$J = \frac{C}{A + B + C}$$

, where

A = the number of click position points in which only trajectory a is present.

B = the number of click position points in which only trajectory b is present.

C = the number of click position points in which both trajectories a and b are present.

The range of the coefficients are between 0 which is incompatible completely between two trajectories, and the number of bifurcation points is zero, and 1 which two trajectories are associated with totally and the number of bifurcation points is the maximum. By contrast, the Jaccard distance calculates the dissimilarity between two trajectories and is equal to the substrate Jaccard similarity coefficient by 1. There is one Jaccard distance example described in figure 24 and table 14.

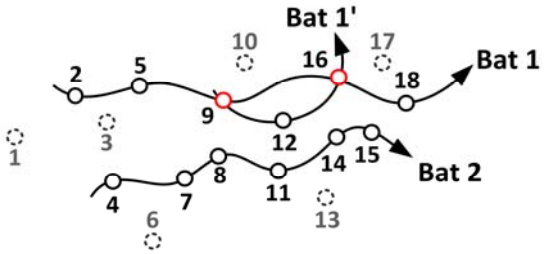


Figure 24. Bifurcation points and trajectory clusters

Table 14. Contingency table of trajectories in figure 24

Data Point Index	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
Bat 1	0	1	0	0	1	0	0	0	1	0	0	0	0	0	0	1	0	1
Bat 1'	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	1	0	0
Bat 2	0	0	0	1	0	0	1	1	0	0	1	0	0	1	1	0	0	0

The Jaccard distance between Bat 1, Bat 1' and Bat 2 are calculated as following:

$$d(\text{Bat 1}, \text{Bat 1'}) = 1 - J = \frac{A + B}{A + B + C} = \frac{4 + 1}{4 + 1 + 2} = 0.714$$

$$d(\text{Bat 1'}, \text{Bat 2}) = 1 - J = \frac{A + B}{A + B + C} = \frac{3 + 6}{3 + 6 + 0} = 1$$

$$d(\text{Bat 2}, \text{Bat 1}) = 1 - J = \frac{A + B}{A + B + C} = \frac{6 + 5}{6 + 5 + 0} = 1$$

Therefore, Bat 1 and Bat 1' are concluded as being in the same trajectory cluster, and Bat 2 is included in another trajectory cluster.

On the other hand, the number of bats (equal to the number of trajectory clusters) recorded in each raw data file is unknown; therefore, the hierarchical clustering method is chosen in our clustering algorithm. After hierarchical clustering, the longest path selection is implemented; if there is more than one longest paths in one trajectory cluster, the path with higher likelihood will be chosen, as shown in figure 25.

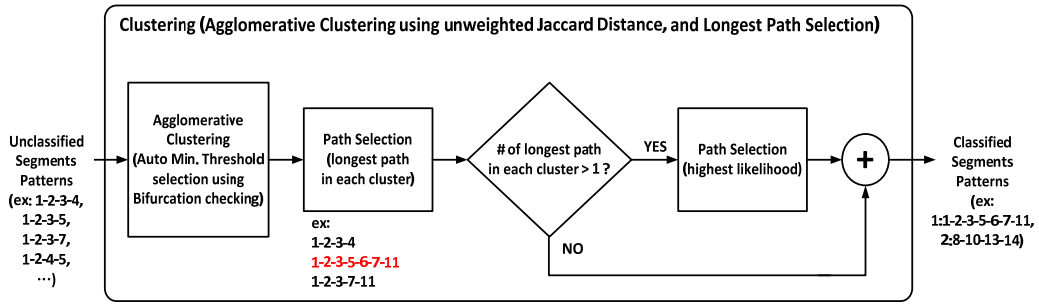


Figure 25. Clustering algorithm

3.5.1 Average Linkage Agglomerative Clustering

An average linkage agglomerative clustering method is utilized in this project. Agglomerative clustering method is more common, much more intuitive and clear in the design phase in the beginning, and has an easier calculation between levels; however, when comparing the number of our target clusters with our clustering samples, a divisive clustering method brings a more efficient computational performance without exhausted recursion (Duda et al., 2001). On the other hand, an average-linkage method is indeed a better method of calculating the distance between groups in this project, as described in the following equation, after comparing the other two distance measurement methods (single-linkage and complete-linkage).

$$d(C_i, C_j) = \frac{1}{n_{C_i} n_{C_j}} \sum_{a \in C_i, b \in C_j} d(C_a, C_b)$$

$$d(C_a, C_b) = 1 - J(a,b)$$

where C_i and C_j are two trajectory clusters, n represents the number of trajectory clustering samples, and J(a, b) represents the Jaccard distance between trajectory a and trajectory b in C_i and C_j respectively.

Figure 26 presents an example of the result of our average linkage agglomerative clustering (Case: 01L471). In this case, there are three individual trajectory clusters classified in the end. Furthermore, we can find that the distribution of each cluster branch is so balanced that it is a characteristic of the average-linkage method.

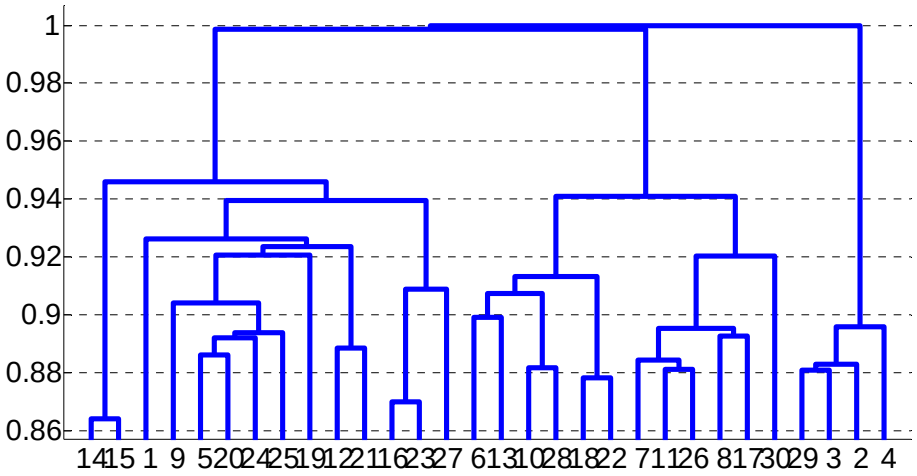


Figure 26. Dendrogram of agglomerative clustering (Case: 01L471)

3.5.2 Automatic Minimum Threshold Selection

A hierarchical clustering method does not need to feed the number of clusters in the beginning as a partitioning clustering method would, but rather it needs to give a threshold, and an end condition to terminate the recursion; Figure 27 describes pseudo codes of the major steps in agglomerative clustering and the threshold is that the current number of clusters ($\hat{c}$) is equal to the target cluster number (c).

Algorithm 4 (Agglomerative hierarchical clustering)

```
1 begin initialize c, ĉ ← n, D_i ← {x_i}, i = 1, ..., n
2         do ĉ ← ĉ − 1
3             Find nearest clusters, say, D_i and D_j
4             Merge D_i and D_j
5         until c = ĉ
6   return c clusters
7 end
```

Figure 27. Major steps in agglomerative clustering

Source: Duda et al., 2001.

However, the number of trajectory clusters in each raw data points file is different, and we

cannot estimate the clustering result of each file manually. Therefore, one minimum threshold selection method using bifurcation checking for determining the end condition of hierarchical clustering is proposed in this project, as shown in Figure 28.

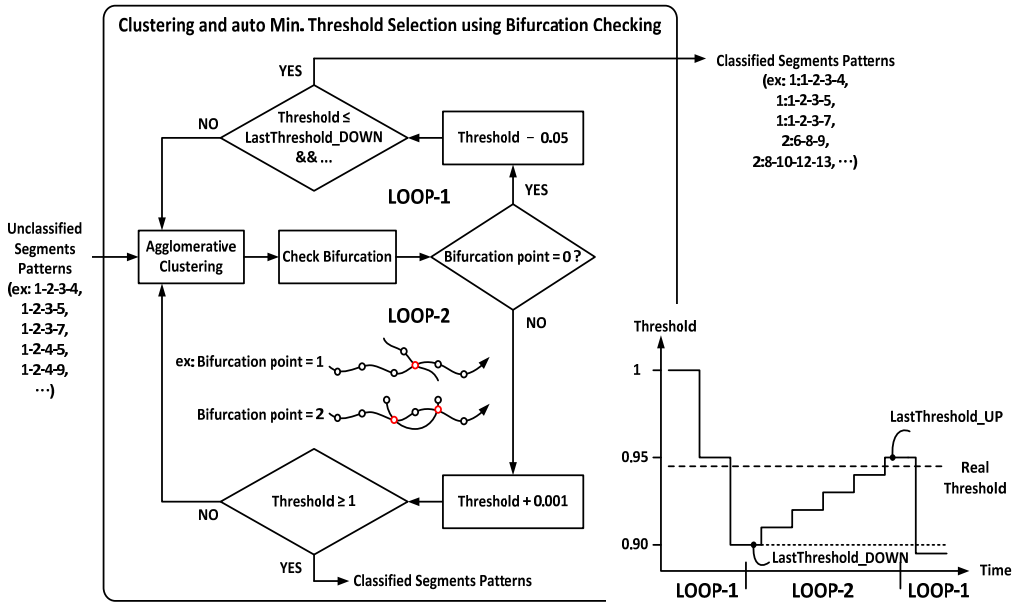


Figure 28. Hierarchical clustering and minimum threshold selection method

In the beginning, the default value of "Threshold" is set to 1, and the values of both "LastThreshold_UP" and "LastThreshold_DOWN" are set to 0. After every hierarchical clustering, a bifurcation measurement will be implemented and the number of bifurcation points will be compared with 0. If the number of bifurcation points is zero, the algorithm will enter "LOOP-1" which the end condition of the clustering "Threshold" will continue to decrease 0.05 each time until the number of bifurcation points is larger than zero. On the other hand, if the number of bifurcation points is not zero, the algorithm will enter "LOOP-2" at which the end condition of the clustering "Threshold" will continue to increase 0.001 each time until the number of bifurcation points equals to zero, and it enters LOOP-1 again. As the current Threshold is larger than the one in LOOP-2 or the current Threshold in LOOP-1 is less than the last term Threshold in LOOP-1 named "LastThreshold_DOWN", this iteration will be terminated and the value of last term Threshold in LOOP-2, named "LastThreshold_DOWN", will be output to be the final end condition of our hierarchical clustering.

Figure 29 presents an example of the result of our minimum threshold selection method in agglomerative clustering (Case: 01L471). In this case, there is a total 50 times of iteration in this threshold selection algorithm; from 3 times of iteration to 29 times are in LOOP-2 and others are in LOOP-1. The LastThreshold_DOWN is 0.9, the LastThreshold_UP is 0.947 and the current Threshold is 0.899. As the current Threshold level (0.899) is less than the level of

LastThreshold_DOWN (0.9), the value of LastThreshold_UP (0.947) will be our final end condition of clustering.

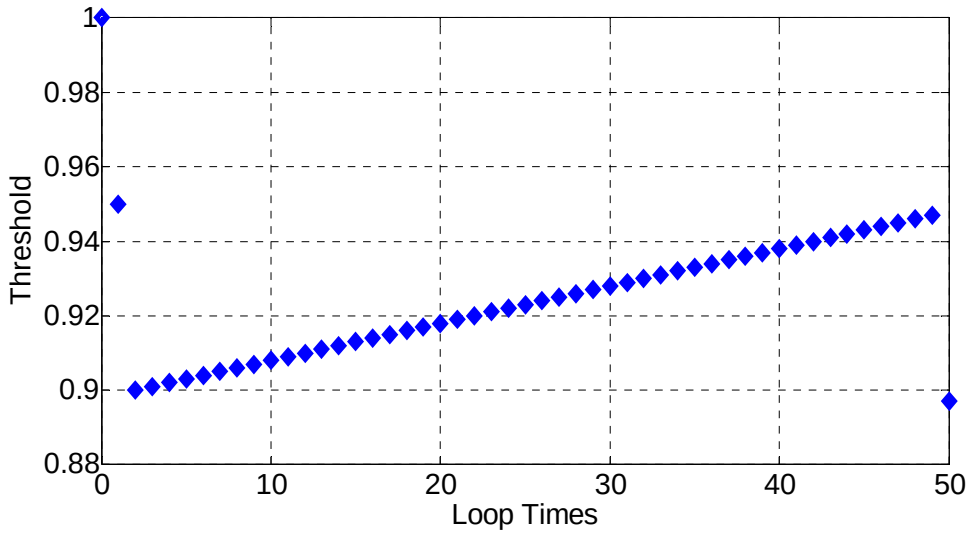


Figure 29. Result of the minimum threshold selection method (Case: 01L471)

This method helps us find the best threshold that guarantees that in clustering the largest number of trajectory clusters will be implemented, and that no bifurcation point will be shared between each cluster. Figure 30 demonstrates that 0.947 is the best threshold level in case 01L471. In case 01L471, there are three trajectories that will be reconstructed.

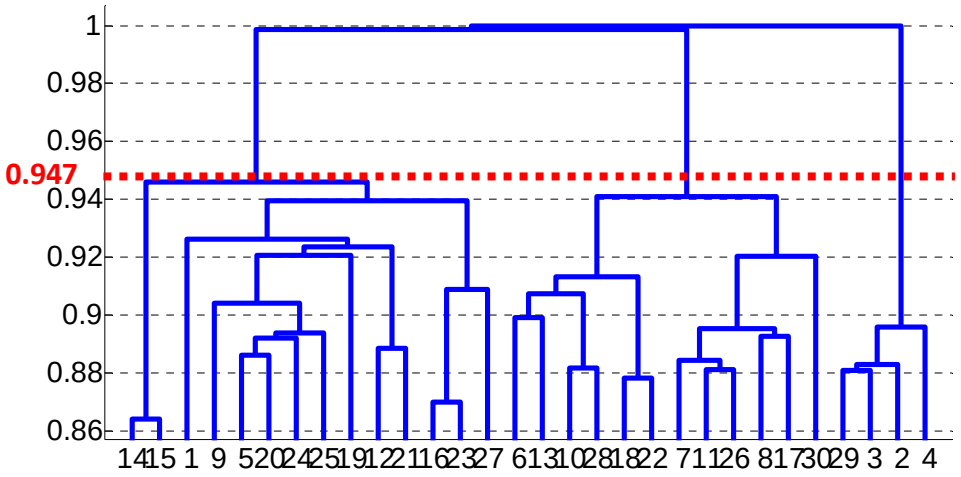


Figure 30. Dendrogram of agglomerative clustering with threshold 0.947 (Case: 01L471)

3.6 Assessment and Preliminary results

In this section three assessment methods and their associated results will be described.

3.6.1 Assessment Method

Reconstructed trajectories will be assessed by the following three factors:

- The ratio (R_{traj}) of the number of trajectories reconstructed by this algorithm from the FLG file that are in the same trajectory group ($n_{reconstructed\_traj \cap FLG\_traj}$) to the number of trajectories from FLG file (n_{FLG_traj}). If the number of concurrent points of trajectories is more than or equal to one, they will belong to the same trajectory group. In other words, as a one click point lying on the curve of trajectory from FLG file is also lying on another trajectory reconstructed by this algorithm, we can say the trajectory from the FLG file is clustered successfully and R_{traj} will increase. The formula of R_{traj} is shown by the following.

$$R_{traj} = \frac{n_{reconstructed\_traj \ \cap \ FLG\_traj}}{n_{FLG\_traj}}$$

- The ratio (R_{click}) of the number of click points which are both recognized by this algorithm and from the FLG file ($n_{reconstructed\_click \cap FLG\_click}$) to the number of click points from the FLG file (n_{FLG_click}). This factor reveals the complete performance of this algorithm, especially for bat flight modelling. The formula of R_{click} is :

$$R_{click} = \frac{n_{reconstructed\_click \ \cap \ FLG\_click}}{n_{FLG\_click}}$$

- The unbiased average deviation (d_{avg_dev}) - the deviation of each reconstructed trajectory curve from its corresponding trajectory curve from the FLG file, is described in Figure 31. This factor also denotes a complete performance of this algorithm especially for the interpolation or approximation method.

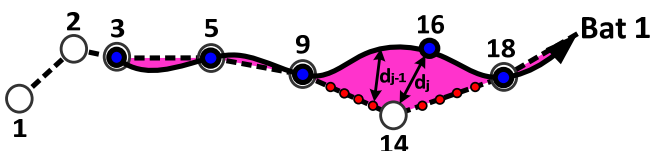


Figure 31. Unbiased average deviation

In figure 31, d represents the distance between one click point or one interpolated point lying on the curve of trajectory reconstructed by this algorithm and the click point from the FLG file with the smallest timing difference; in other words, all distances between the 3D points of the nearest temporal stamps on corresponding trajectories will be measured, summed and averaged. This is shown here:

$$d_{avg\_dev} = \sum_{i} \sum_{j=1}^{n} \frac{d_j}{n}$$

where i represents the indexes of overlapped segments which share both the beginning and end click points; n equals to the number of time-domain linear interpolation points within each segment of the trajectory from the FLG file.

After the three assessment factors defined above are applied, two further types of assessment will be implemented – the FLG-trajectory test pattern assessment which chooses well-established manually-reconstructed trajectory patterns to be the input patterns of this algorithm; and real raw data reconstruction assessment, as shown in figure 32. The former assessment uses test patterns which are so clean without any noise and outliers; therefore, ideally we can undertake the simplest operation process, and get the best reconstructing result; furthermore, unit level performance can be monitored and evaluated in detail. On the other hand, the latter assessment involves the input raw data being mixed with lots of noise and outliers; therefore, the ability of filtering, segment selection and the speed of data processing is the major observation point in this kind of real case operation. Furthermore, the results of the real case assessment and reconstructed trajectory curves will be reviewed and assessed by biologists with a thorough consideration. After all, the target of this algorithm is to provide both an efficient and objective method of data analysis for biologists.

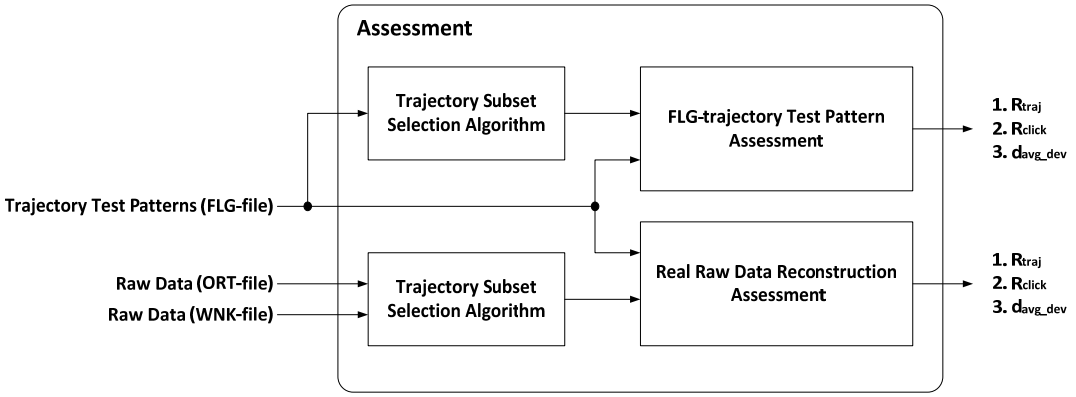
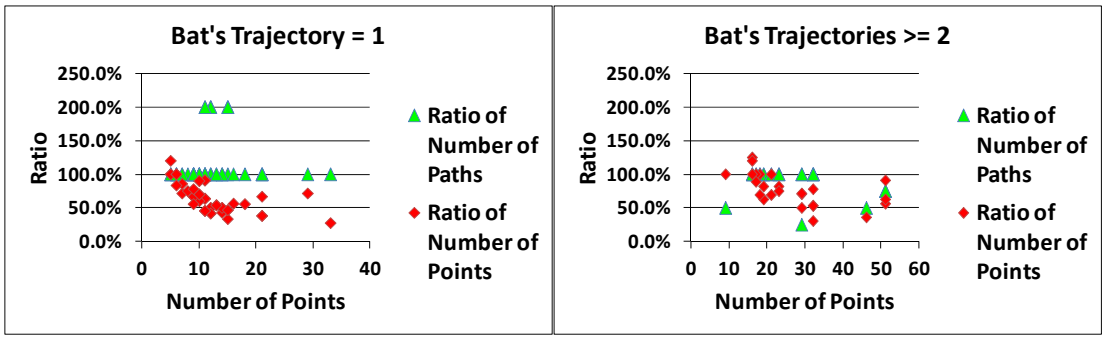


Figure 32. Assessment methods

3.6.2 Preliminary Results

In this session, FLG-trajectory test patterns are utilized to assess the algorithm's function and performance. Approximately 50 FLG files are selected randomly which include both one bat trajectory and multiple bat trajectories. Results are shown in figure 33, 36 and 39.



(a) (b)

Figure 33. (a) R_{traj} and R_{click} to number of points from one-trajectory files; (b) R_{traj} and R_{click} to number of points from multiple-trajectories files

In figure 33 (a), there are three single trajectories are recognized to three double section trajectories (R_{traj} is 200%); each trajectory is segmented into two short trajectories as shown in figure 34. In figure 34, the dash line is the hand-reconstructed trajectory, and the two colour lines are segmented trajectories reconstructed by our algorithm. In our algorithm, the top 1% possible trajectories are permitted to keep branching; however, the data points in the FLG files are much simplified and efficient. Therefore, some trajectory's segment subsets in this test bench may always be pruned like case 30L382.

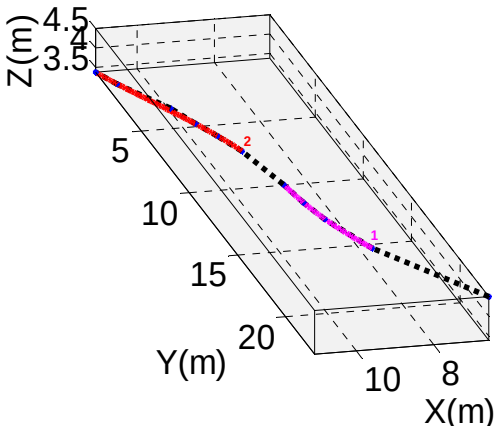


Figure 34. Trajectory reconstruction from FLG file 30L382

On the other hand, in figure 33 (b), six trajectories are missing compared with the hand-reconstructed trajectories (R_{traj} is less than 100%); these trajectories are grouped

with other trajectories and then filtered out by the longest path selection method in our algorithm. Figure 35 reveals the phenomenon in which there is a difficulty in thoroughly reconstructing tightly crisscrossing trajectories in this project.

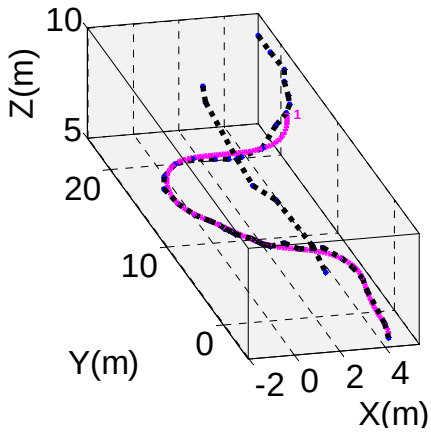
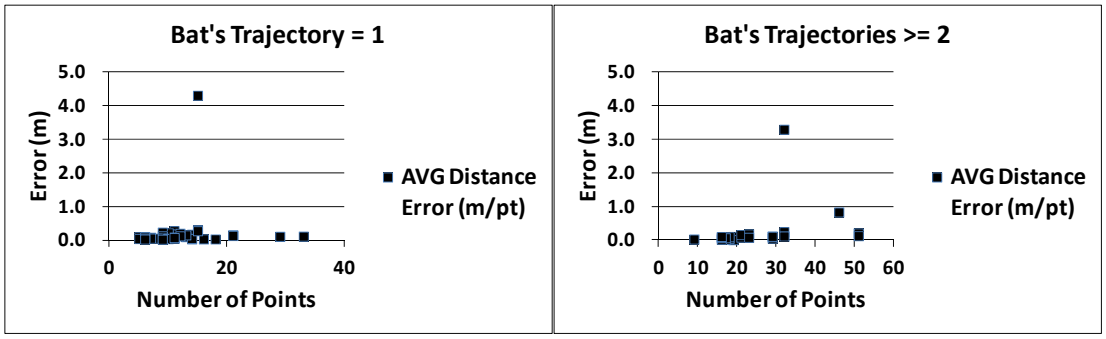


Figure 35. Trajectory reconstruction from FLG file 30L394

Besides above phenomena, we can also notice that R_{click} decreases gradually as the number of data points in the one trajectory file increases. The reason is that in the computation of such small samples, the rate of utilization of data points in the more clusters' case is better than that in the fewer clusters' case; in other words, lots of candidate segments in the fewer clusters' case will be screened out.



(a) (b)

Figure 36. (a) d_{avg_dev} to the number of points from one-trajectory files; (b) d_{avg_dev} to the number of points from multiple-trajectories files

In Figure 36, most curves fit hand-reconstructed trajectories well, of which their point-to-point distance deviations are less than 1 meter. Nevertheless, there are two cases that gain our attention - case 01L453 and case 01M071 as shown in Figure 37 and 38 respectively. Because there is no ground truth can judge the results, experts in the biological field can help us to check them. After all, finding out the difference from subjective manual observation is another objective of this project, besides saving manpower.

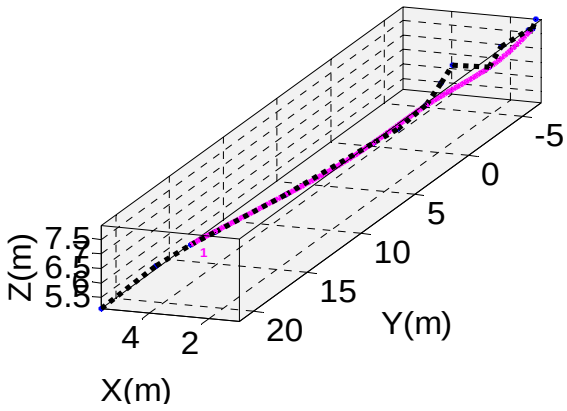


Figure 37. Trajectory reconstruction from FLG file 01L453

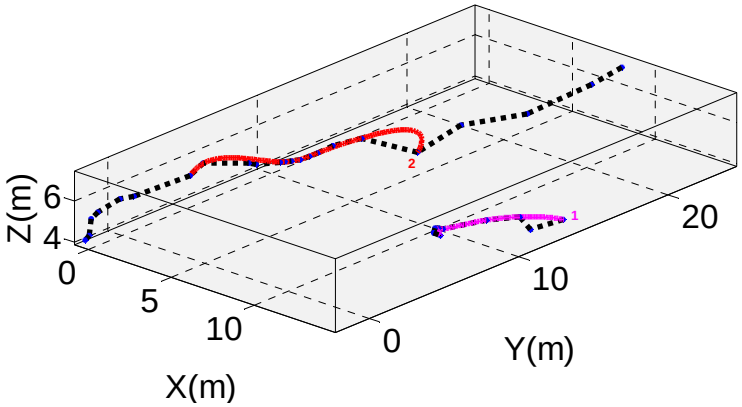
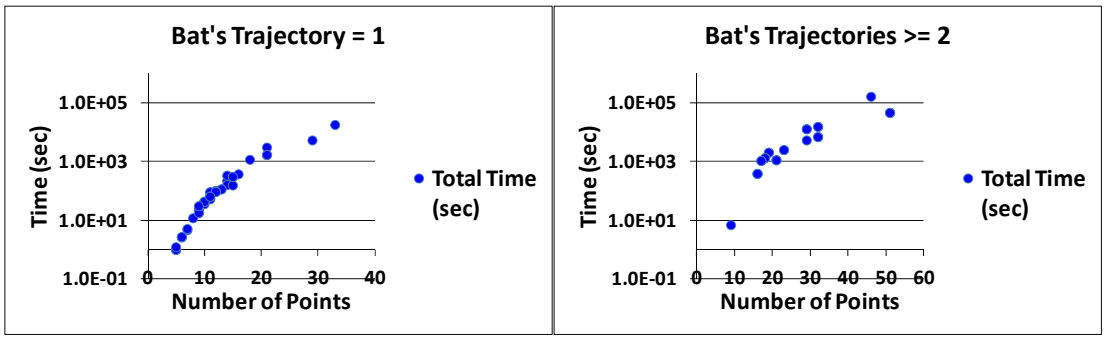


Figure 38. Trajectory reconstruction from FLG file 01M071



(a) (b)

Figure 39. (a) Computational time to number of points from one-trajectory files; (b) computational time to number of points from multiple-trajectories files

Figure 39 reveals the most difficult issue of our algorithm – the large computational time. Reconstructing approximately 50 data points takes about 100,000 seconds. It will be a significant problem when implementing real raw data reconstruction. Therefore, speeding up computational time is another important task of this project, and will be described in detail in the next chapter.

4. Optimization

Two methodologies are proposed here for solving the crucial problem of the long operation time. An adaptive interpolation points selection method is proposed to achieve both high speed and high resolution interpolation calculation in curvature measurement; an importance sampling method is utilized by looking at the 3 point segments table to achieve the most efficient branching.

4.1 Adaptive Interpolation Points Selection

Cubic spline interpolation is utilized in this project to approximate the original flight curve. However, the shape of reconstructed curve is not always regular and fixed, but varied according to certain parameters - the ratio of any adjacent segments' lengths, the angle between any adjacent segments and the number of cubic spline interpolation points, as shown in figure 40 and 41.

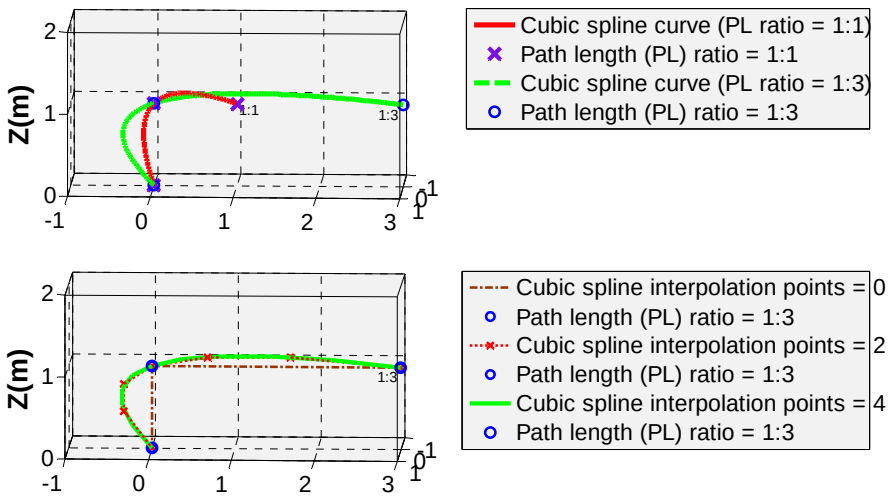


Figure 40. Influences of cubic spline curve by path length ratio of adjacent segments and the number of interpolation points (angle is 90 degree)

In figure 40, the figure describes the shape of cubic spline curve which is determined by the different ratios of path lengths of the adjacent segments. If the ratio is small, the shape of spline curve is mild; on the other hand, if the ratio is large, the shape curves markedly. In contrast, in the below figure, the shape of the cubic spline curve is determined by different numbers of interpolation points; the more interpolation points, the smoother and more accurate the spline curve.

Figure 41 describes the shape of cubic spline curve which is determined by both the angle and the ratio of the path lengths of adjacent segments. If the angle is blunt, the shape of the spline curve is straighter; on the other hand, if the ratio is sharp, the shape curves markedly.

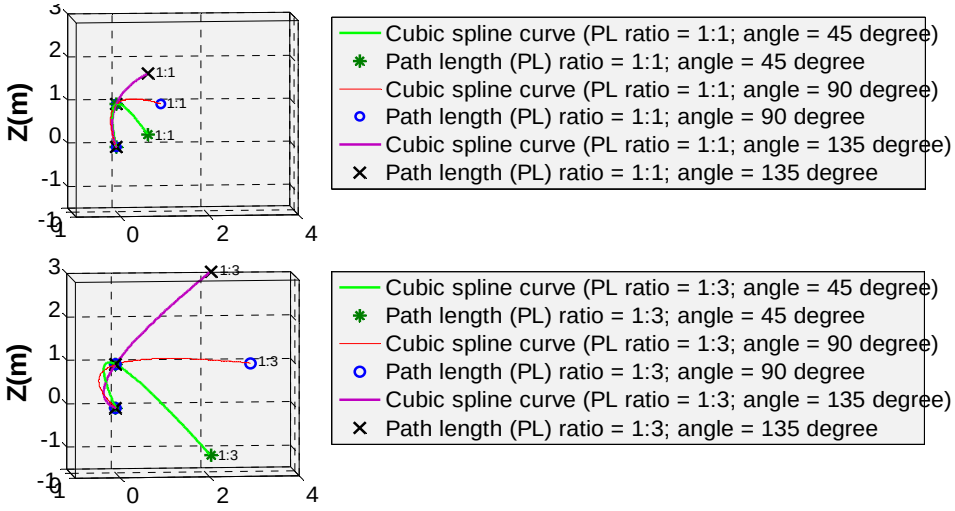


Figure 41. Influences of cubic spline curve by angle and path length ratio of adjacent segments (the number of interpolation points is 100)

These interpolated flight curves affect the results of the bat flight parameters measurement, especially for path length and curvature; the former also influences the velocity indirectly. Therefore, how to choose the ideal number of interpolation points becomes an important topic; after all, the number of interpolation points is the only parameter which can be controlled because the path length and angle of adjacent segments are relative to the click positions.

We know that a larger number of interpolation points provides more accuracy, but it takes more computational time. Therefore, an adaptive interpolation points selection mechanism is the ideal solution with respect to the blind selection of a large number of points. The adaptive interpolation points selection method chooses a sufficient number of points and keeps its performance similar to select a larger number of points, and additionally saves a great deal of time.

Figure 42 describes the influences on path length and curvature by the number of interpolation points, where the horizontal axis represents the number of interpolation points and the vertical axis represents the influences on path length (top) and curvature (bottom) by a different number of interpolation points compared with the influences when the number of interpolation points equals 1000; besides, the different ratio of lengths of adjacent segments also brings different results. We can find that the accuracy of curvature

measurement is about 60% of 1000 interpolation points chosen, if the ratio of adjacent segments' lengths is 1:20 and we choose 20 interpolation points. Also, the accuracy of curvature measurement is about 80% of 1000 interpolation points chosen if the ratio of adjacent segments' lengths is 1:40 and we choose 60 interpolation points.

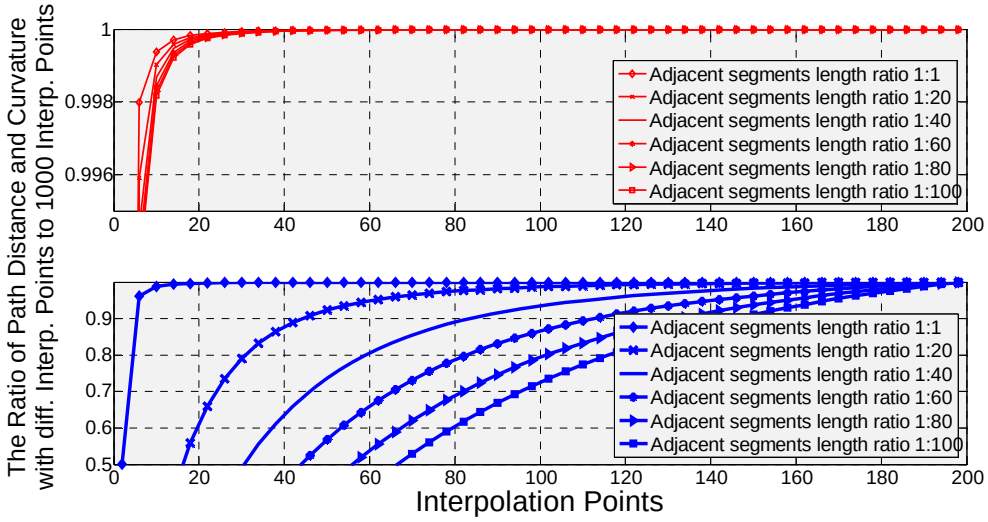


Figure 42. Influences of path distance and curvature by the number of interpolation points

Because path length is influenced by the adjacent segments' length ratio and angle, and the influences of curvature by angle are not obvious, this adaptive selection mechanism will only be utilized in curvature measurement where the number of interpolation points will be selected by monitoring the adjacent segments' length; in other flight parameters' measurement, the number of interpolation points will be set to 10, as shown in figure 43.

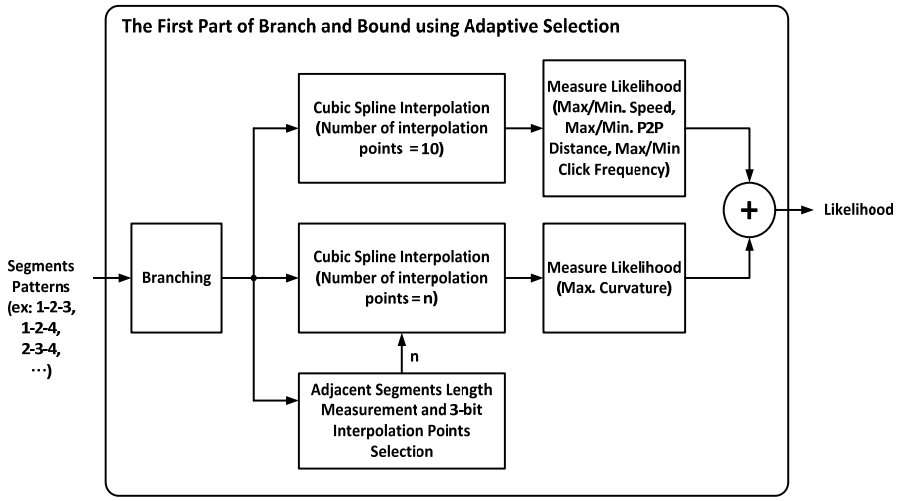


Figure 43. The first part of a branch-and-bound algorithm using an adaptive interpolation points selection method

The pseudo code is described in Table 15.

Table 15. Pseudo code of an adaptive interpolation points selection method

- Load X/Y/Z points : X, Y and Z
- For i = 2 to length of X/Y/Z points -1
  - Measure length ratio of adjacent segments :
  - A = [(X(i+1)-X(i)) (Y(i+1)-Y(i)) (Z(i+1)-Z(i))], B = [(X(i-1)-X(i)) (Y(i-1)-Y(i)) (Z(i-1)-Z(i))]
  - Calculate ratio : R(i-1) = norm(A)/norm(B)
- End For
- If [max(R) <= 5] OR [(min(R) <= 1/5]
  - n = 10
- Else if [max(R) <= 10 & max(R) > 5] OR [min(R) <= 1/10 & min(R) > 1/5]
  - n = 20
- Else if [max(R) <= 20 & max(R) > 10] OR [min(R) <= 1/20 & min(R) > 1/10]
  - n = 35
- Else if [(max(R) <= 30 & max(R) > 20)] OR [(min(R) <= 1/30 & min(R) > 1/20)]
  - n = 50
- Else if [(max(R) <= 40 & max(R) > 30)] OR [(min(R) <= 1/40 & min(R) > 1/30)]
  - n = 65
- Else if [(max(R) <= 50 & max(R) > 40)] OR [(min(R) <= 1/50 & min(R) > 1/40)]
  - n = 80
- Else if [(max(R) <= 60 & max(R) > 50)] OR [(min(R) <= 1/60 & min(R) > 1/50)]
  - n = 95
- Else
  - n = 110 (Best interpolation resolution)
- End If

In case 30L474, there are approximately 141,700 segments calculated for curvature measurement in the process of trajectory reconstruction; however, there are only 4,448 segments of which their ratios of adjacent segments' lengths are larger than 5, as shown in figure 44. It means approximately 70% of the calculation time for curvature measurement can be saved by using this selection method (best interpolation resolution = 110).

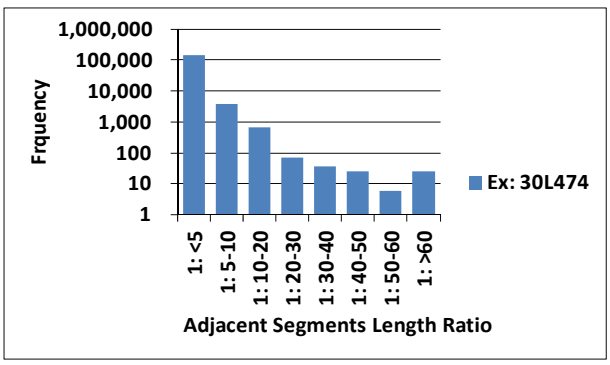


Figure 44. Accumulated adjacent segments length ratio (Case: 30L474)

4.2 Importance Sampling

In the process of trajectory reconstruction, the process of branch-and-bound takes most of the computational time (approximately over 95%) and bat flight parameters' measurement is in the majority. Therefore, efficient branching is another important task of optimization.

In our algorithm, each 3 point segment has already been measured and filtered by checking their likelihoods with the bat flight models at the start of the process. Some 3 point segments are selected for branch-and-bound, and others are deleted. Those remaining 3 point segments are called trajectory candidates, which means that they are regarded as more important than the others and have a higher probability of being a part of the complete trajectory. Therefore, a reliable way to decrease the number of branching is to delete those new extended segments that are not listed on the candidates' table - a list of those remaining 3 point segments. It means that each new branched trajectory in each recursion run will be given a likelihood weighting - 0's or 1's - and this likelihood is not calculated with its neighbours, despite its parent samples – the 3 point segments table. Of course, a few good branched trajectories may be over killed for this kind of efficiency improvement, as shown in Figure 45.

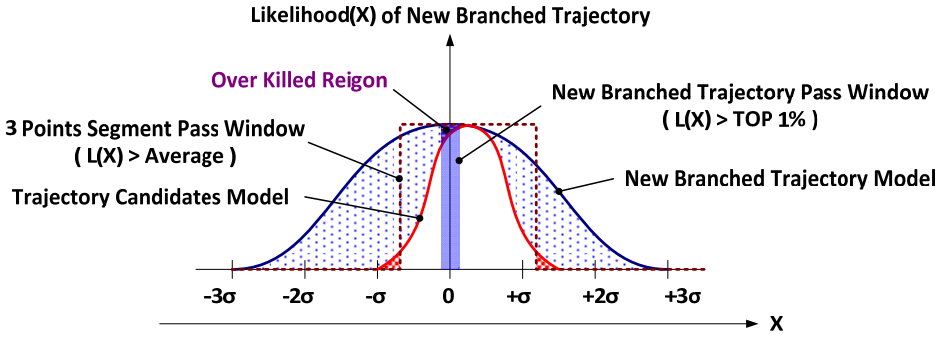


Figure 45. Trajectory candidates, new branched trajectory models and over killed region

The shape of the cubic spline segment curve $P_k(t)$ is determined by the position and tangent vectors of both the beginning point, P_k and P_k', and the end point, P_{k+1} and P_{k+1}' as shown in figure 46.

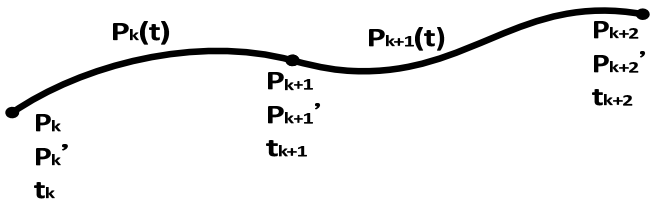


Figure 46. Multiple piecewise cubic spline segments curve

However, for determining the tangent factor of the inner joint P_{k+1}' between any two cubic spline segments Pk and Pk+1, the following equation needs to be solved (Rogers and Adams, 1990):

$$t_{k+2}P'_k + 2(t_{k+1}t_{k+2})P'_{k+1} + t_{k+1}P'_{k+2} = \frac{3}{t_{k+1}t_{k+2}}\left[t^2_{k+1}(P_{k+2} - P_{k+1}) + t^2_{k+2}(P_{k+1} - P_k)\right]$$

It means that the shape of the former segment curve $P_k(t)$ between P_k and P_{k+1} will be affected by the position of the next point P_{k+2}. Therefore, the likelihood of any 3 point segment listed on the trajectory candidates table will not be totally the same as any new extended 3 point segment within the new branched trajectories because the tangent vectors in the former segment are never the same as their corresponding tangent vectors in the latter segment. Fortunately, from computer graphics theory, we know that the influences of the beginning and stop tangent vectors are much less important than their positions as shown in Figure 47, where the blending function of the cubic spline segment curve Pk(t) is described as follows (Rogers and Adams, 1990):

$$P_k(t) = \begin{bmatrix} F_1(t) & F_2(t) & F_3(t) & F_4(t) \end{bmatrix} \begin{bmatrix} P_k \\ P_{k+1} \\ P'_k \\ P'_{k+1} \end{bmatrix}$$

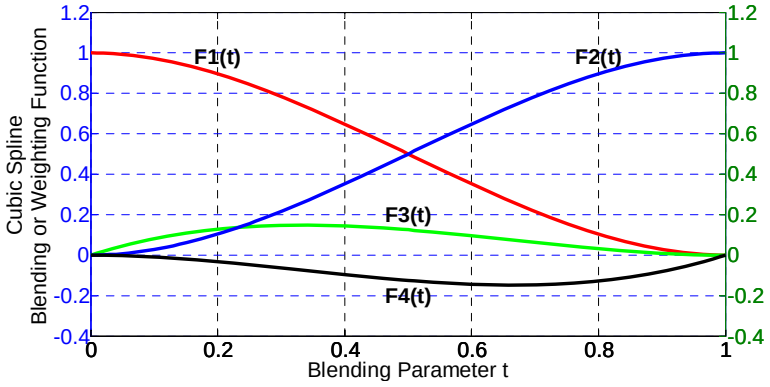


Figure 47. Cubic spline blending or weighting function

Any point lying on the curve $P_k(t)$ is a weighted sum of above four factors. In figure 47, if F1(0) = 1 and F2(0) = F3(0) = F4(0) = 0, this curve passes through the point P_1. Similarly, if F1(1) = 0, F2(1) = 1 and F3(1) = F4(1) = 0, this curve passes through the point P_2.

4.3 Extensions to the basic framework

An improved branch-and-bound algorithm with adaptive interpolation points selection and importance sampling (look-up trajectory candidates table) methods is shown in figure 48.

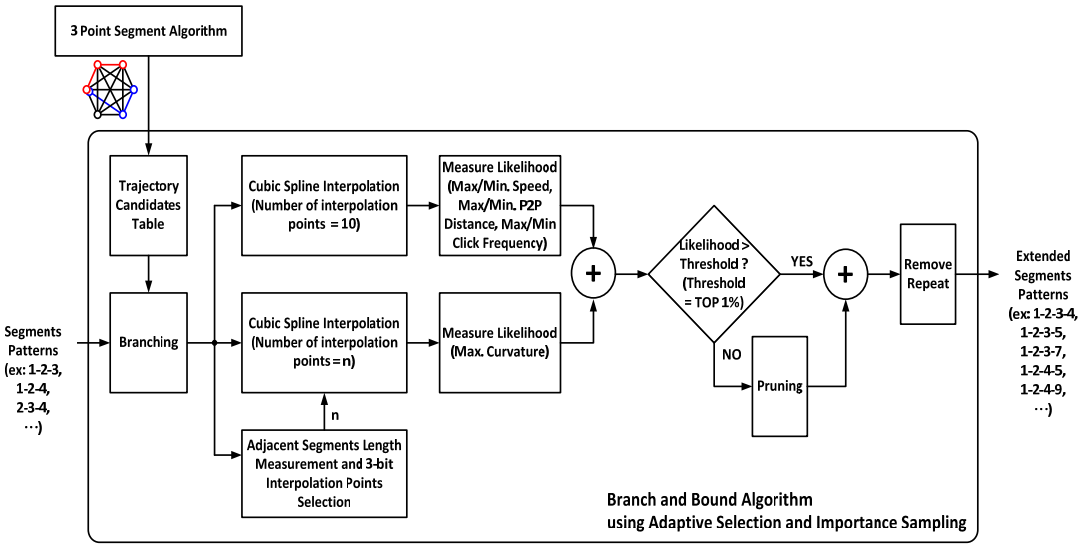


Figure 48. Branch-and-bound algorithm using adaptive selection and importance sampling

In this session, FLG-trajectory test patterns with multiple bat's trajectories from FLG files are utilized to assess the optimized algorithm's function and performance. Comparison results between the original algorithm and the optimized algorithm on the assessment factors of R_{traj}, R_{click}, d_{avg_dev} and the total computational time are described on figures 49 to 52.

In figure 49, most results of R_{traj} between the two algorithms are similar, except case 01L471 (51 points), and this will be discussed below. Additionally, the average R_{traj} on the original algorithm is 85%; on the optimized algorithm it is 91%.

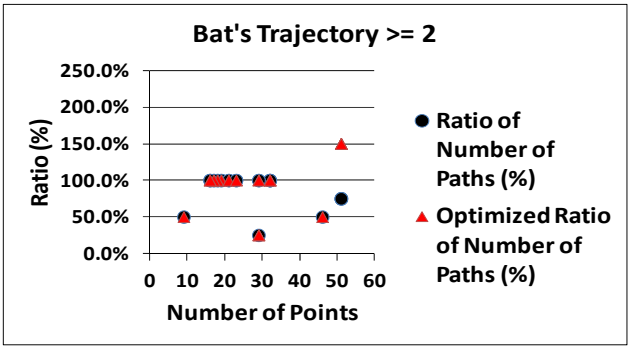
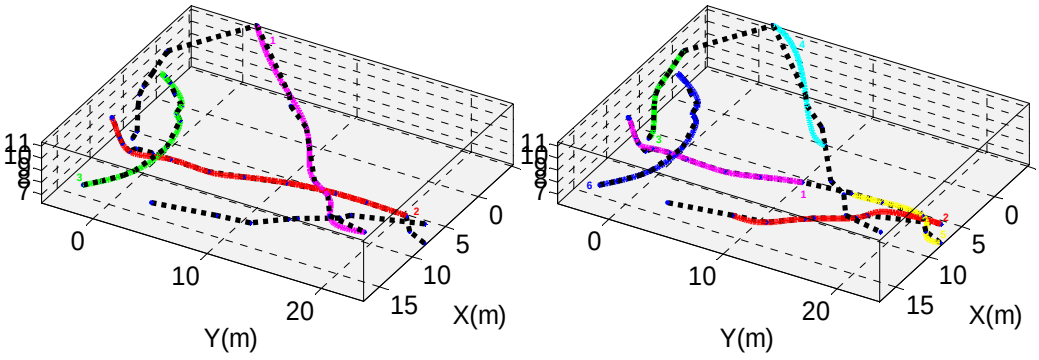


Figure 49. Rtraj to the number of points from multiple-trajectories files

Figure 50 shows trajectories reconstructed by the original and optimized algorithms respectively in case 01L471 where the dash curves are hand-reconstructed trajectories and the coloured curves are trajectories reconstructed by the study's algorithms. Some trajectories in figure 50 (a) are segmented into several segments after optimization as shown in figure 50 (b) because of the effect of the importance sampling method and because the data points in the FLG-trajectory test pattern are simplified. In contrast, some missing trajectories shown in figure 50 (a) are successfully reconstructed after optimization as shown in figure 50 (b) because of the effect of the adaptive interpolation points selection method.



(a) (b)

Figure 50. Trajectory reconstruction from FLG file 01L471.
(a) original algorithm; (b) optimized algorithm

In figure 51, the results for the assessment factor R_{click} between the two algorithms are similar. The average R_{click} on the original algorithm is 77%; on the optimized algorithm it is 69%.

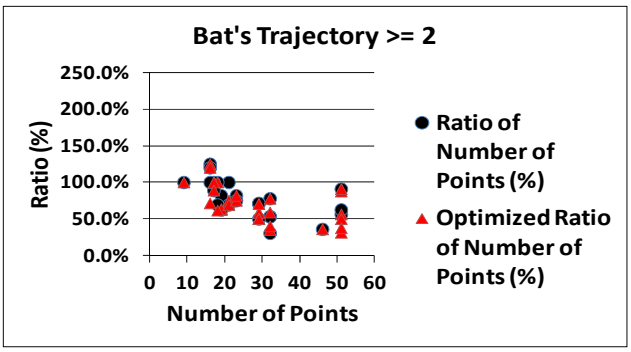


Figure 51. Rclick to the number of points from multiple-trajectories files

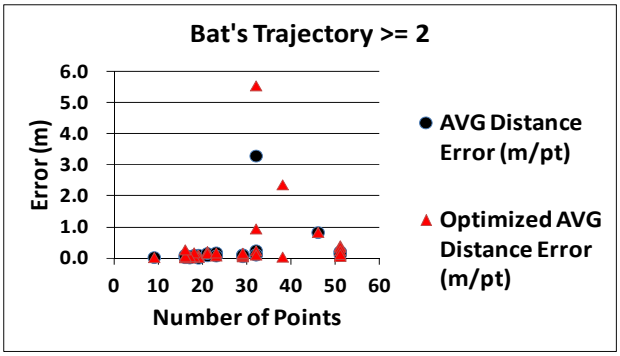
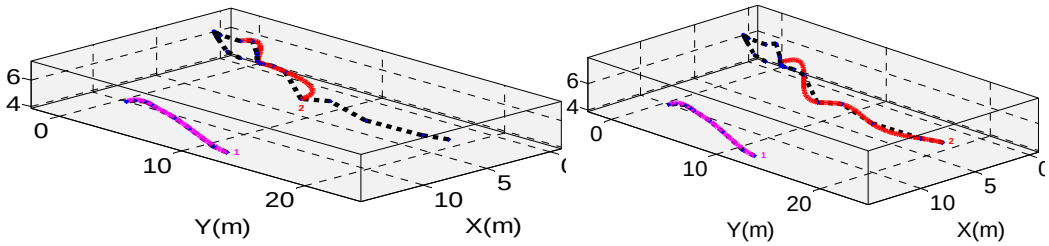


Figure 52. davg_dev to number of points from multiple-trajectories files

In figure 52, the majority of assessment results for d_{avg_dev} between the two algorithms are similar, except for case 01M071 (32 points) , and this will be discussed below. The average d_{avg_dev} on the original algorithm is 0.27 m; on the optimized algorithm it is 0.69 m.

Case 01M071's value for the assessment factor d_{avg_dev} is large, and has been highlighted before, as shown in figure 38; we know that a large value for the assessment result doesn't mean that the trajectory reconstructed by our algorithm is poor, but instead reveals that there is a large difference between the predictions from the biologist and the statistical bat flight models utilized by our algorithm. Further more, if we review the trajectories in case 01M071 carefully, we can find that two optimization methods help the trajectory's characteristics more accurately replicate the bat's behaviour in figure 53 (b); be better than Figure 53 (a), although d_{avg_dev} from the optimized algorithm in figure 53 (b) is larger than the original algorithm in figure 53 (a).



(a) (b)

Figure 53. Trajectory reconstruction from FLG file 01M071.
(a) original algorithm; (b) optimized algorithm

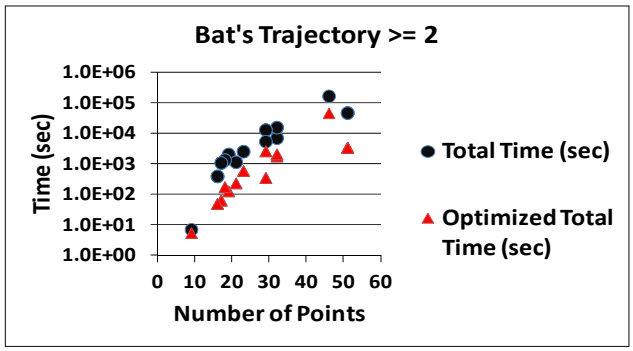


Figure 54. Computational time to the number of points from multiple-trajectories files

Figure 54 reveals a considerably improved speed for the computational time by utilizing the optimized algorithm. The average speed improvement is about 9 times faster for multiple-trajectories files on the FLG-trajectory test pattern assessment. The average computational time on the original algorithm it is 20,064.33 seconds; on the optimized algorithm it is 4,369.81 seconds.

On the other hand, for real raw data reconstruction assessment, the comparison of case 30L474 (32 data points)'s computational times on different branch-and-bound optimizations is shown in figure 55. If the number of interpolation points is 100, the speed improvement between the complete branching and the importance sampling is about 2.5 times faster. Moreover, the total improvement between the oldest and latest branch-and-bound mechanisms is approximately 9 times faster. This speed improvement mainly depends on the number of data points.

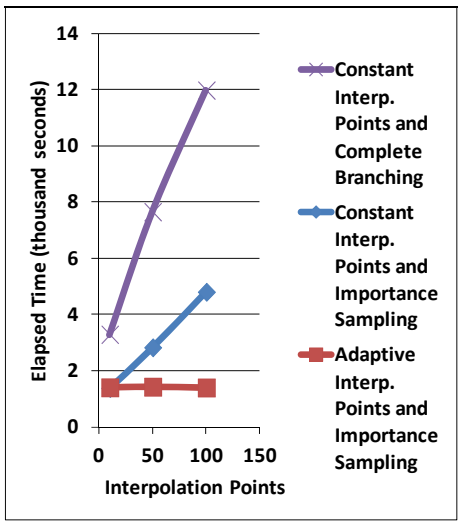
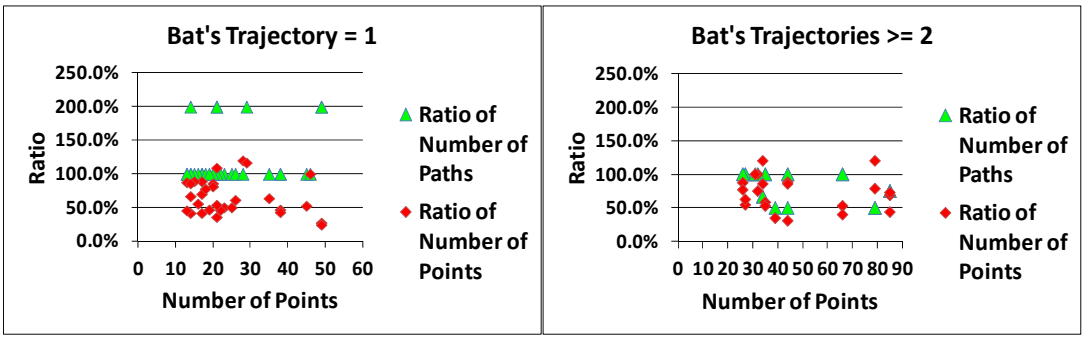


Figure 55. Comparison of branch-and-bound computational time (Case 30L474)

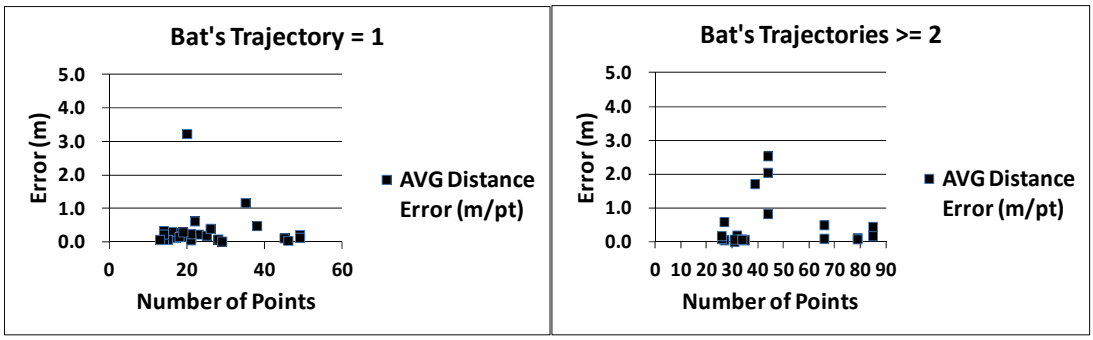
5. Results

Optimized algorithms have been evaluated by real raw data reconstruction assessment in this chapter, as shown in figures 56, 57 and 58.



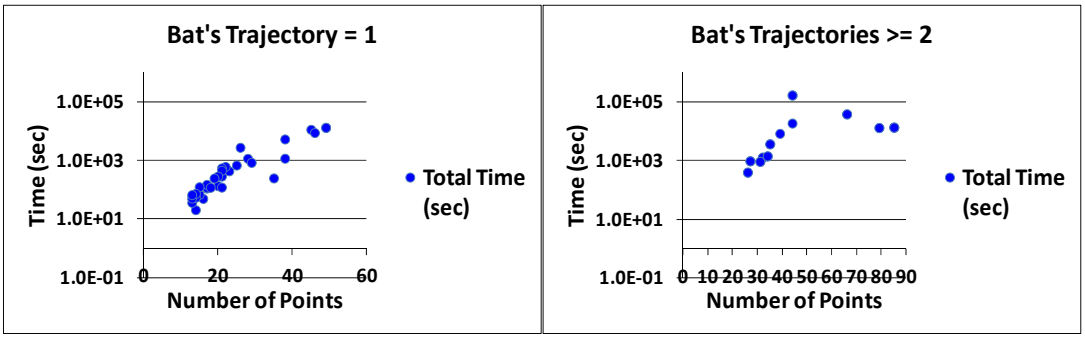
(a) (b)

Figure 56. (a) R_{traj} and R_{click} ratios to the number of points from one-trajectory files; (b) R_{traj} and R_{click} ratios to the number of points from multiple-trajectories files



(a) (b)

Figure 57. (a) d_{avg_dev} average distance error to the number of points from one-trajectory files; (b) d_{avg_dev} average distance error to the number of points from multiple-trajectories files



(a) (b)

Figure 58. (a) Computational time to the number of points from one-trajectory files; (b) computational time to the number of points from multiple-trajectories files

Table 16 describes the comparison of assessment results with the different algorithms and assessment methods. Besides the improvement in computational time from the original to the optimized algorithm as discussed before, with the increased data points and noise (echoes and outliers), the system's complexity has generally risen (the average computational time per point has risen more than 60% from 102.62 to 165.19), but the results of the other two assessment factors (R_{traj} and R_{click}) have remained similar (R_{traj} has decreased just 12.26% and R_{click} has increased only 6.07%). Furthermore, d_{avg_dev} has improved markedly (deviation has decreased 34.78%). This proves that the performance of prior flight models adopted in the branch-and-bound algorithm is reliable.

Table 16. Comparison of assessment results, algorithms, and assessment methods

Bat =1 / Bat >= 2	Average ratio (R_{traj}) of the number of trajectories (%)	Average ratio (R_{click}) of the number of click points (%)	Average unbiased average deviation (d_{avg_dev}) (m)	Average computational time per point (second/pt)
Original algorithm assessed by utilizing FLG-trajectory test patterns	109.38 / 84.62	65.64 / 77.02	0.2 / 0.27	32.12 / 481.86
Optimized algorithm assessed by utilizing FLG-trajectory test patterns	N/A / 91	N/A / 69	N/A / 0.69	N/A / 102.62
Optimized algorithm assessed by utilizing real raw data files	117.24 / 81.06	65.44 / 73.46	0.56 / 0.45	33.93 / 165.19

Furthermore, as discussed before, the data points in the FLG-trajectory test pattern are simplified and the segments are reconstructed restrictedly; the usage of importance sampling will worsen this phenomenon (R_{click} has decreased from 77.02% to 69%). Nevertheless, this phenomenon will be much improved in terms of real raw data reconstruction assessment (R_{click} has increased to 73.46%). There are some potential bat click points and segments hidden by noises and ignored by the biologist in the process of manual reconstruction. Our model based branch-and bound algorithm utilizing the maximum

likelihood theory can restore these, as shown in figures 59 and 60. However, still some errors of prediction may happen, as shown in figures 61 and 62, and it has decreased the rate of success of trajectory recognition (R_{traj} has decreased from 91 to 81.06 as shown in table 16).

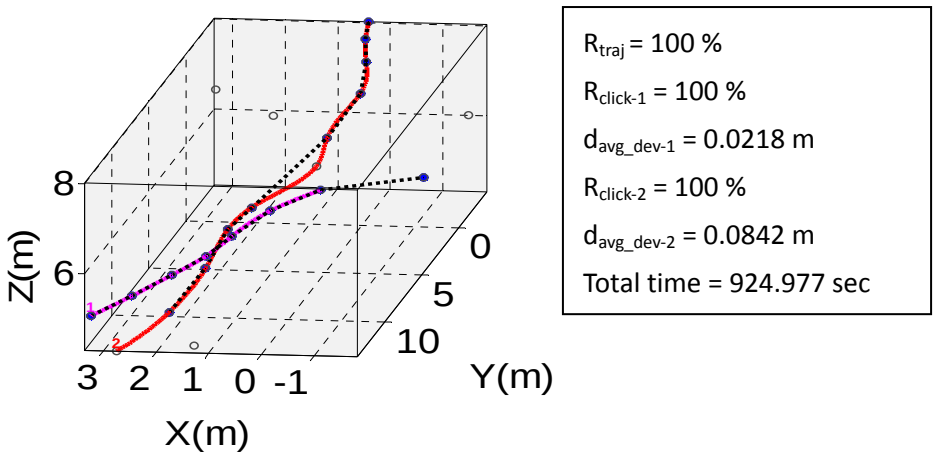


Figure 59. Path-2 is reconstructed successful by potential point and increases $R_{click-2}$ to 100% (Case 29L441)

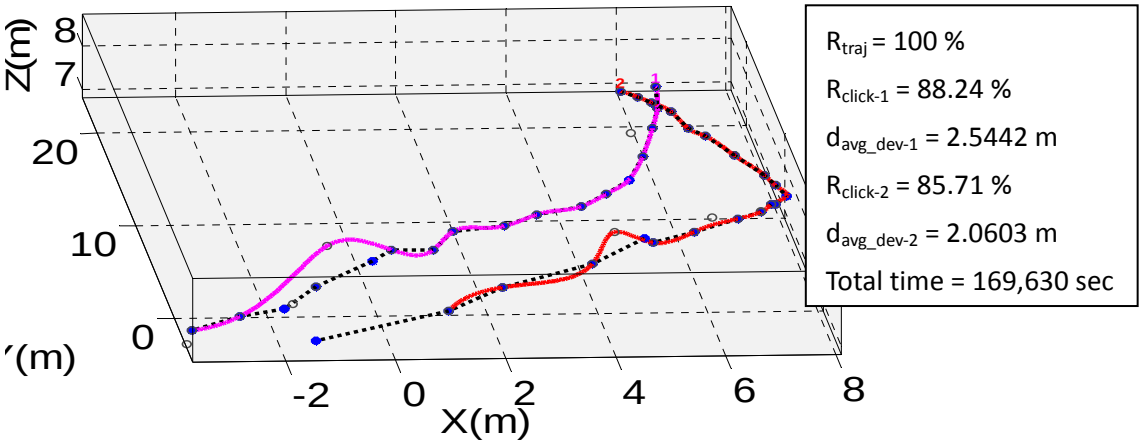


Figure 60. Both path-1 and -2 are reconstructed successful by potential points and increase $R_{click-1\ and\ -2}$ to 88.24% and 85.71% respectively (Case 30L492)

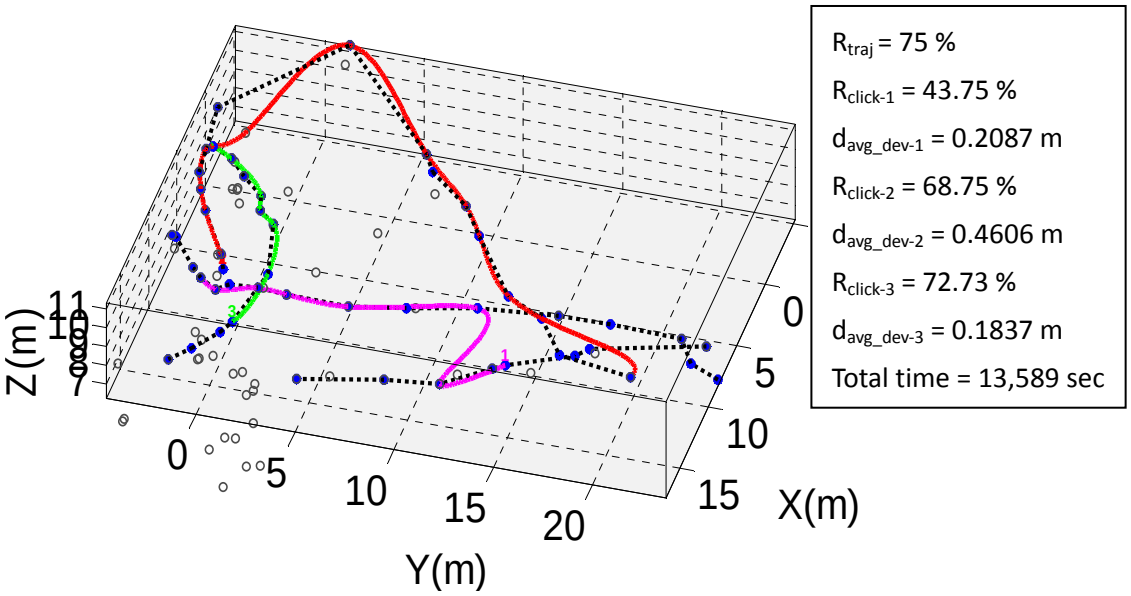


Figure 61. Path-1 is a prediction error and decreases Rtraj to 75% (Case 01L471)

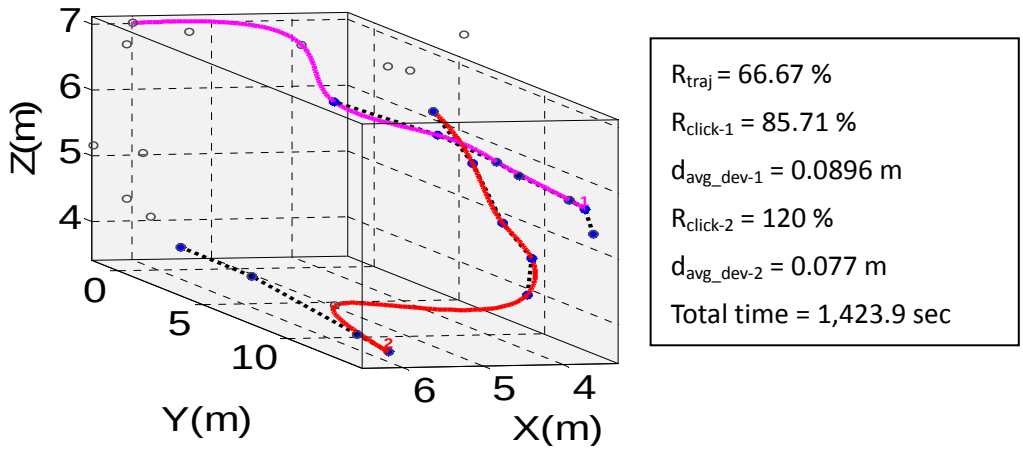


Figure 62. Path-2 is a prediction error and decreases Rtraj to 66.67% (Case 29L481)

Using this algorithm, some unexpected trajectories that are hard to predict manually are constructed. With the biologist's help, some unexpected trajectories are judged to be accurate, but some are not. The former can help the biologist gain a great deal of valid and new research material, such as case 29L511 and 01L571 as shown in Figures 63 and 64.

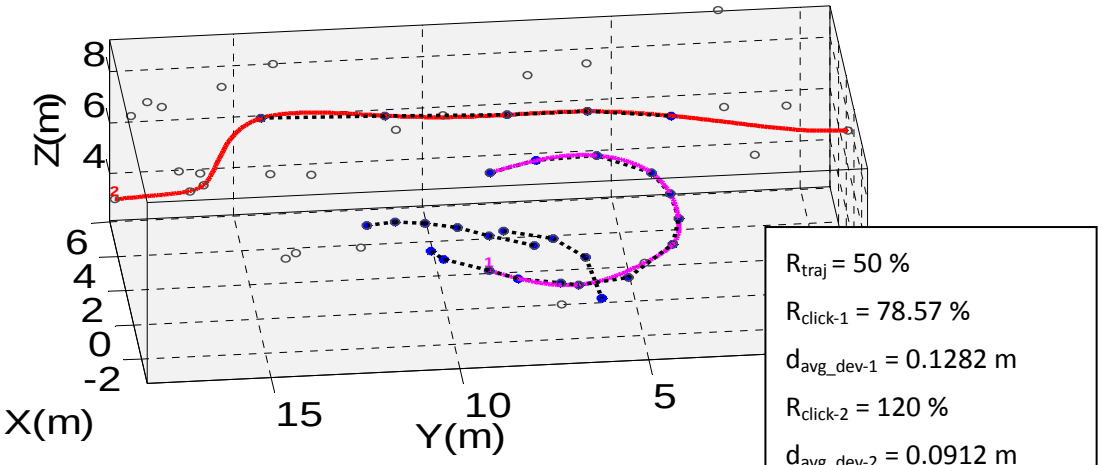


Figure 63. Path-2 is unexpected and matches bat beha

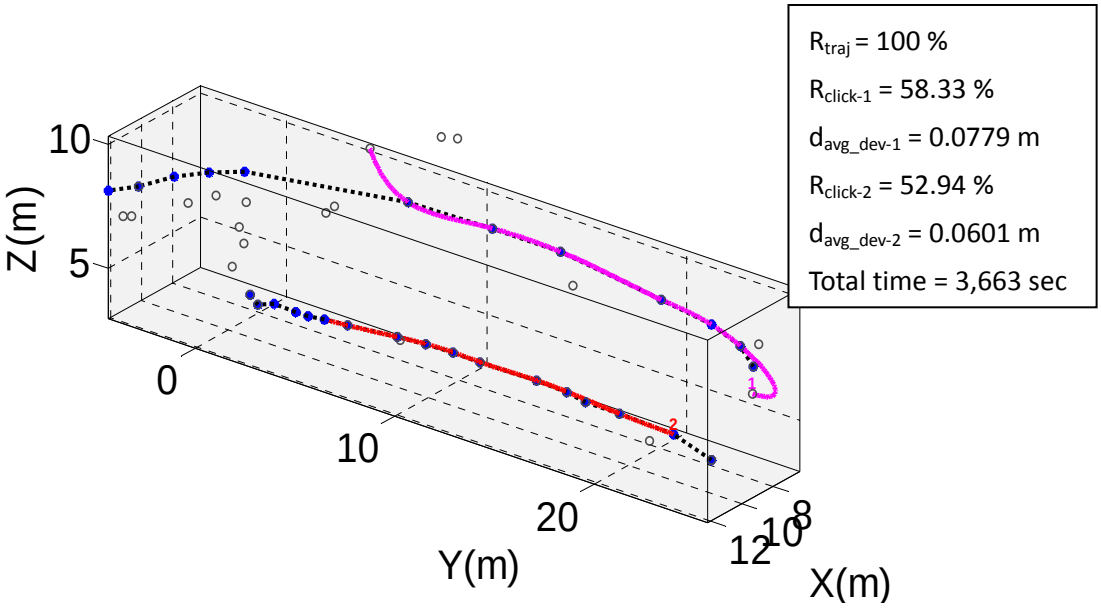


Figure 64. Path-1 is unexpected and matches bat behaviour (Case 01L571)

On the other hand, the unreliable trajectories expose the weakness of this algorithm, especially for the prior flight models. It reveals that our current prior flight model cannot cover all bat flight behaviours and additionally cannot filter out some paths that do not match the bat's biological behaviour, such as cases 01L531, 01M041, 29L473 and 29L421, as shown in Figures 65 to 68 respectively. Based on a bat's biological behaviour, some flight features can be added and some can be weighted; of course, this new prior flight model would be more complicated and may need the researcher to cope with the crucial problem of large computational time again.

Finally, there are four test rules suggested by biologist and described as follows:

A) How many click points support a turn? If there is only one or two, it may be untrue.

B) How many click points support a direction change? Long curves without much support are

unlikely to be true.

C) How curved is it? A highly curved path is often unrealistic for an aerial hawking species such as *Nyctalus leisleri.*

D) Are there many height changes? Manoeuvrability is mainly on the horizontal. Drastic height changes need excellent support by many points.

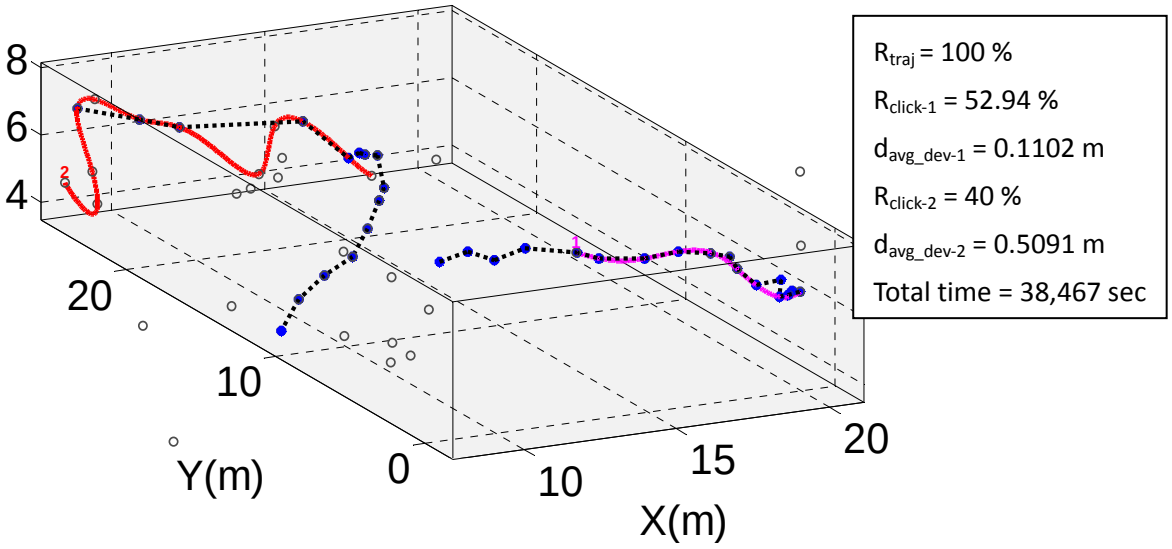


Figure 65. Path-2 is unexpected and doesn't match bat behaviour (Case 01L531)

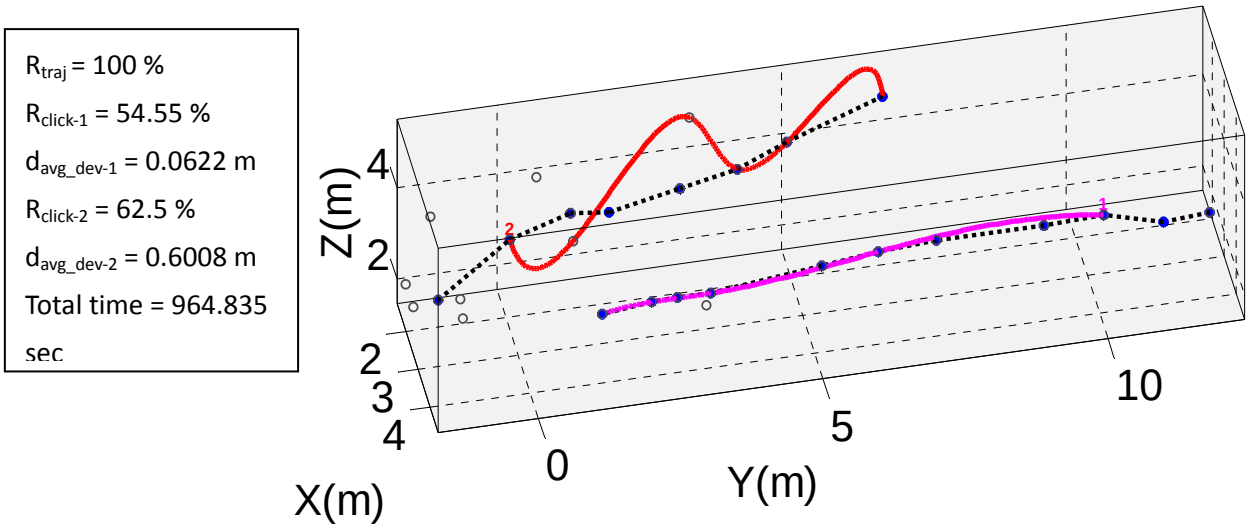


Figure 66. Path-2 is unexpected and doesn't match bat behaviour (Case 01M041)

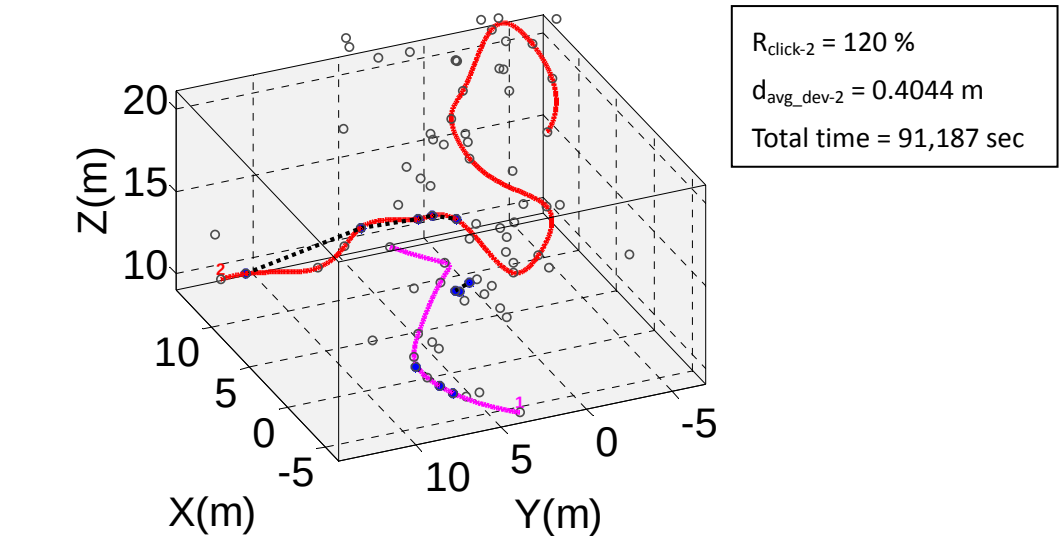


Figure 67. Path-2 is unexpected and doesn't match bat behaviour (Case 29L473)

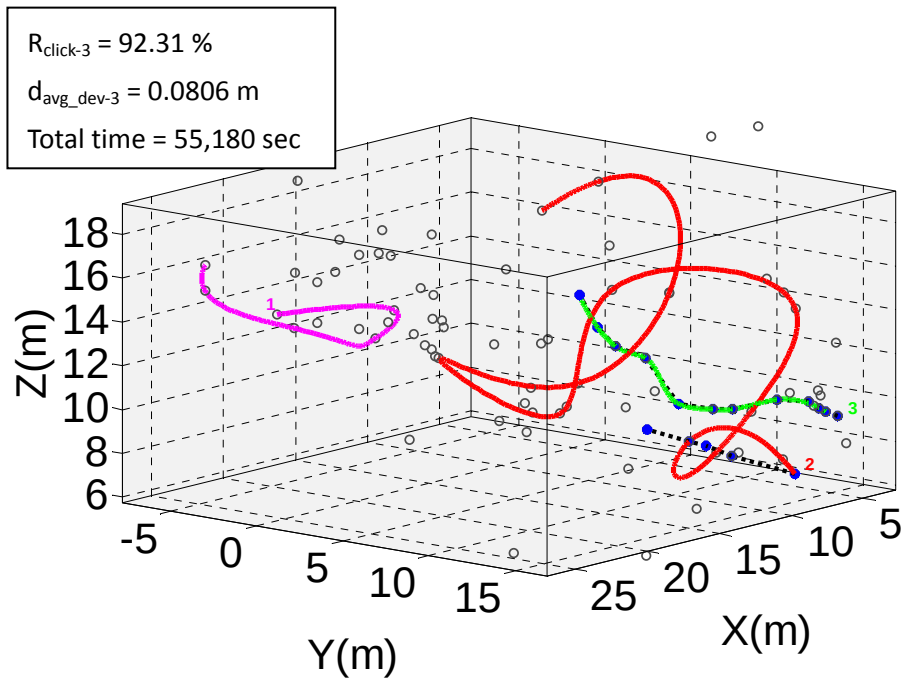


Figure 68. Path-2 is unexpected and does not match bat behaviour (Case 29L421)

6. Conclusion and Discussion

The recording of flight paths and vocalizations is important to gain a complete understanding of the behaviour of bats, a most diverse group of mammals. Additionally, the study and reproduction of a bat's detection system, an innate ability which has evolved over 50 million years, is also valuable in multiple fields. Moreover, it is valuable to understand just how Computer Science can play an important role in such cross-disciplinary work in the real world.

The aim of this project has been to research and present a framework that can reconstruct 3D flight trajectories from multiple bat call positions automatically, as shown by a system overview of this project in Figure 1. Prior flight models were adopted into a trajectory's segment subset selection algorithm, utilizing unsupervised classification and branch-and-bound methods for recognizing bat flight segments and clustering bat trajectories in this project. Moreover, two optimization methodologies, adaptive interpolation points selection and importance sampling, were proposed, and were found to really improve that most crucial problem, the large computational time, by approximately 5 to 10 times; they also enhanced the accuracy of the interpolation at the same time.

Average reconstruction rates for bat trajectories in each real raw data file corresponding to any FLG-file were about 81-118%; average recognition rates for bat call positions in each real raw data file corresponding to each trajectory in any FLG-file were about 65-74%; the average unbiased distance deviation of each trajectory between trajectories in any real raw data file and any FLG-file were about 0.45-0.56 (m/pt). Finally average computational times are approximately 34-165 seconds respectively. These assessment results prove the reliability and efficiency of this project. Not only saving manpower, this project provides a repeatable and objective way of data analysis.

This project therefore appears to provide a feasible methodology to reconstruct bat trajectories; however, there are still some difficulties that need to be compensated for. First is the need for a more precise and complete prior flight model which defines bat biological behaviour in detail and can cover most bat trajectories. Besides, because of tightly crisscrossing trajectories or bats chasing each other, these are difficult to classify by current clusters in this project; therefore, a more reliable clustering method is a challenge for this job in the future. Finally, bat species' recognition and classification is another interesting task that could be approached.

Bibliography

Akins, J. B., Kennedy, M. L., Schnell, G. D., Sanchez-Hernandez, C., Romero-Almaraz, M. L., Wooten, M. C. & Best, T. L. (2007). 'Flight Speeds of Three Species of Neotropical Bats: Glossophaga Soricina, Natalus Stramineus, and Carollia Subrufa', *Acta Chiropterologica*, Vol. 9, No. 2, pp. 477-482.

Allee, W. C. (1927). 'Animal Aggregations', *The Quarterly Review of Biology*, Vol. 2, No. 3, pp. 367-398.

Balleri, A., Griffiths, H. D., Woodbridge, K., Baker, C. J. & Holderied, M. W. (2010[a]). 'Bat-Inspired Ultrasound Tomography in Air', *2010 IEEE Radar Conference*, pp. 44-47.

Balleri, A., Griffiths, H., Holderied, M. & Baker, C. (2010[b]). 'Bat-inspired Multi-Harmonic Waveforms', *2010 International Waveform Diversity and Design Conference (WDD)*, pp. 86-89.

Balleri, A., Woodbridge, K., Baker, C. J. & Holderied, M. W. (2009). 'Classification of Flowers By Bats: Comparison with the Radar Case', *2009 International Waveform Diversity and Design Conference*, pp. 1-3.

Bell, G. P. & Fenton, M. B. (1984). 'The Use of Doppler-Shifted Echoes as a Flutter Detection and Clutter Rejection System - The Echolocation and Feeding-Behavior of Hipposideros-Ruber (Chiroptera, Hipposideridae)', *Behavioral Ecology and Sociobiology*, Vol. 15, No. 2, pp. 109-114.

Bertini, M., Del Bimbo, A. & Seidenari, L. (2012). 'Multi-Scale and Real-Time Non-Parametric Approach for Anomaly Detection and Localization', *Computer Vision and Image Understanding*, Vol. 116, No. 3, pp. 320-329.

Calderara, S., Alaimo, C., Prati, A. & Cucchiara, R. (2009). 'A Real-Time System for Abnormal Path Detection', *Crime Detection and Prevention (ICDP 2009), 3rd International Conference*, pp. 1-6.

Cheetham, A. H. & Hazel, J. E. (1969) 'Binary (Presence-Absence) Similarity Coefficients', *Journal of Paleontology*, Vol. 43, No. 5, pp. 1130-1136.

Comaniciu, D., Ramesh, V. & Meer, P. (2000). 'Real-Time Tracking of Non-Rigid Objects Using Mean Shift', *IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2000)*, pp. 142-149.

Corcoran, A. J., Barber, J. R., Hristov, N. I. & Conner, W. E. (2011). 'How Do Tiger Moths Jam Bat Sonar?', *Journal of Experimental Biology*, Vol. 214, No. 14, pp. 2416-2425.

Crandall, M. G. & Rabinowitz, P. H. (1971). 'Bifurcation from Simple Eigenvalues', Journal of Functional Analysis, Vol. 8, No. 2, pp. 321-340.

Denzinger, A., Siemers, B. M., Schaub, A. & Schnitzler, H. U. (2001). 'Echolocation by The Barbastelle Bat, Barbastella Barbastellus', *Journal of Comparative Physiology a-Sensory Neural and Behavioral Physiology*, Vol. 187, No. 7, pp. 521-528.

Duda, R. O., Hart, P. E. & Stork, D. G. (2001). *Pattern Classification*. 2nd Edition. Canada: John Wiley & Sons.

Eastman, K. M. & Simmons, J. A. (2005). 'A Method of Flight Path and Chirp Pattern Reconstruction for Multiple Flying Bats', *Acoustics Research Letters Online-Arlo*, Vol. 6, No. 4, pp. 257-262.

Erwin, H. R., Wilson, W. W. & Moss, C. F. (2001). 'A Computational Sensorimotor Model of Bat Echolocation', *Journal of the Acoustical Society of America*, Vol. 110, No. 2, pp. 1176-1187.

Faria, D. R. & Dias, J. (2009). '3D Hand Trajectory Segmentation by Curvatures and Hand Orientation for Classification Through a Probabilistic Approach', *Intelligent Robots and Systems*, *IROS 2009. IEEE/RSJ International Conference*, pp. 1284-1289.

Gaisler, J. (1979). 'Results of Bat Census in a Town Mammalia Chiroptera', *Vestnik Ceskoslovenske Spolecnosti Zoologicke*, Vol. 43, No. 1, pp. 7-21.

Garcia, J., Molina, J. M., Besada, J. A. & de Miguel, G. (2007). 'Model-Based Trajectory Reconstruction Using IMM Smoothing and Motion Pattern Identification', *Information Fusion*, *10th International Conference*, pp. 1-8.

Garcia, J., Soto, A., de Miguel, G., Besada, J. & Tarrio, P. (2008). 'Trajectory Reconstruction Techniques for Evaluation of ATC Systems', *Digital Communications - Enhanced Surveillance of Aircraft and Vehicles*, *TIWDC/ESAV 2008. Tyrrhenian International Workshop*, pp. 1-6.

Ghose, K. & Moss, C. F. (2003). 'The Sonar Beam Pattern of a Flying Bat as It Tracks Tethered Insects', *Journal of the Acoustical Society of America*, Vol. 114, No. 2, pp. 1120-1131.

Ghose, K., Zotkin, D., Duraiswami, R. & Moss, C. F. (2001). 'Multimodal Localization of a Flying Bat', *IEEE International Conference on Acoustics*, *Speech*, *and Signal Processing*, pp. 3057-3060.

Goerlitz, H. R., ter Hofstede, H. M., Zeale, M. R. K., Jones, G. & Holderied, M. W. (2010). 'An Aerial-Hawking Bat Uses Stealth Echolocation to Counter Moth Hearing', *Current Biology*, Vol. 20, No. 17, pp. 1568-1572.

Griffin, D. R. (1958). *Listening in the Dark*, New Haven: Yale University Press.

Grodzinski, U., Spiegel, O., Korine, C. & Holderied, M. W. (2009). 'Context-Dependent Flight Speed: Evidence for Energetically Optimal Flight Speed in the Bat Pipistrellus Kuhlii?', *Journal of Animal Ecology*, Vol. 78, No. 3, pp. 540-548.

Hagino, T., Hiryu, S., Fujioka, S., Riquimaroux, H. & Watanabe, Y. (2007). 'Adaptive SONAR Sounds by Echolocating Bats', *2007 Symposium on Underwater Technology and Workshop on Scientific Use of Submarine Cables and Related Technologies*, Vol. 1&2, pp. 647-651.

Hayward, B. & Davis, R. (1964). 'Flight Speeds in Western Bats', *Journal of Mammalogy*, Vol. 45, No. 2, pp. 236-242.

Hiryu, S., Hagino, T., Shiort, Y., Riquiniaroux, H. & Watanabe, Y. (2007). 'Compensation Behaviors in Echolocating Bats Measured by a Telemetry Microphone During Flight', *2007 Symposium on Underwater Technology and Workshop on Scientific Use of Submarine Cables and Related Technologies*, Vol. 1 & 2, pp. 578-582.

Holderied, M. W., Baker, C. J., Vespe, M. & Jones, G. (2008). 'Understanding Signal Design During the Pursuit of Aerial Insects by Echolocating Bats: Tools and Applications', *Integrative and Comparative Biology*, Vol. 48, No. 1, pp. 74-84.

Hooper, J. H. D. (1981). 'The Use of an Ultrasonic Receiver to Obtain Distribution Data for Pipistrelles and Other Bats within the London Area', *The London Naturalist: the journal of the London Natural History Society*, No. 60, pp. 47-63.

Hopkins, H. L., Sanchez-Hernandez, C., Romero-Almaraz, M. D., Gilley, L. M., Schnell, G. D. & Kennedy, M. L. (2003). 'Flight Speeds of Four Species of Neotropical Bats', *Southwestern Naturalist*, Vol. 48, No. 4, pp. 711-714.

Hurtado, M. & Nehorai, A. (2008). 'Bat-inspired Adaptive Design of Waveform and Trajectory for Radar', *42nd Asilomar Conference on Signals*, *Systems and Computers*, pp. 36-40.

Jain, A. K. & Dubes, R. C. (1988). *Algorithms for Clustering Data*, *Prentice-Hall Advanced Reference Series*. Upper Saddle River, NJ: Prentice-Hall.

Jain, A. K., Murty, M. N. & Flynn, P. J. (1999). 'Data Clustering: A Review', *ACM Computing Surveys*, Vol. 31, No. 3, pp. 264-323.

Jensen, M. E., Moss, C. F. & Surlykke, A. (2005). 'Echolocating Bats Can Use Acoustic Landmarks for Spatial Orientation', *Journal of Experimental Biology*, Vol. 208, No. 23, pp. 4399-4410.

Kalman, R. E. (1960). 'A New Approach to Linear Filtering and Prediction Problems', *Transactions of the ASME*, *Journal of Basic Engineering*, No. 82, Series D, pp. 35-45.

Kanzawa, Y. & Oishi, S. (1999). 'Calculating Bifurcation Points with Guaranteed Accuracy ', *IEICE Trans Fundam Electron Commun Comput Sci (Inst Electron Inf Commun Eng)*, Vol. E82–A, No. 6, pp. 1055-1061.

Kline, M. (1998). *Calculus: An Intuitive and Physical Approach*. N. Y: Dover.

Lee, E. Y., Betke, M. & Kunz, T. H. (2004). 'Bats in Motion: Stereo Object Recognition and Trajectory Analysis of Flying Bats', *Bat Research News*, Vol. 45, No. 4, pp. 236-252.

Maina, J. N. (2000). 'What It Takes to Fly: The Structural and Functional Respiratory Refinements in Birds and Bats', *Journal of Experimental Biology*, Vol. 203, No. 20, pp. 3045-3064.

Mao, J. & Jain, A. K. (1996). 'A Self-Organizing Network for Hyperellipsoidal Clustering (HEC)', IEEE Transactions on Neural Networks, Vol. 7, No. 1, pp. 16–29.

Mills, R. S., Barrett, G. W. & Farrell, M. P. (1975). 'Population Dynamics of the Big Brown Bat (Eptesicus fuscus) in Southwestern Ohio', *Journal of Mammalogy*, Vol. 56, No. 3, 591-604.

Moss, C. F. & Surlykke, A. (2010). 'Probing the Natural Scene by Echolocation in Bats', *Frontiers in Behavioral Neuroscience*, Vol. 4, No. 33, pp. 1-16.

Mueller, H. & Emlen, J. (1957). 'Homing in Bats', *Science*, Vol. 126, No. 3268, pp. 307-308.

Narendra, P. M. & Fukunaga, K. (1977). 'A Branch and Bound Algorithm for Feature Subset Selection', *IEEE Transactions on Computers*, Vol. C-26, No. 9, pp. 917-922.

Obrist, M. K., Rathey, E., Bontadina, F., Martinoli, A., Conedera, M., Christe, P. & Moretti, M. (2011). 'Response of Bat Species to Sylvo-Pastoral Abandonment', *Forest Ecology and Management*, Vol. 261, No. 3, pp. 789-798.

Riskin, D. K. (2001). 'Pipistrellus Bodenheimeri', *Mammalian Species*, No. 651, pp. 1-3.

Rogers, D. F. & Adams, J. A. (1990). Mathematical Elements for Computer Graphics. 2nd Edition. London: McGraw-Hill.

Russ, J. M., Briffa, M. & Montgomery, W. I. (2003). 'Seasonal Patterns in Activity and Habitat Use by Bats (Pipistrellus Spp. and Nyctalus Leisleri) in Northern Ireland, Determined Using a Driven Transect', *Journal of Zoology*, Vol. 259, pp. 289-299.

Russo, D. & Jones, G. (2002). 'Identification of Twenty-Two Bat Species (Mammalia: Chiroptera) from Italy by Analysis of Time-Expanded Recordings of Echolocation Calls', *Journal of Zoology*, Vol. 258, pp. 91-103.

Russo, D., Cistrone, L. & Jones, G. (2005). 'Spatial and Temporal Patterns of Roost Use by Tree-Dwelling Barbastelle Bats Barbastella Barbastellus', *Ecography*, Vol. 28, No. 6, pp. 769-776.

Sanchez-Hernandez, C., Romero-Almaraz, M. D. L., Wooten, M. C., Schnell, G. D. & Kennedy, M. L. (2006). 'Speed in Flight of Common Vampire Bats (Desmodus Rotuadus)', *Southwestern Naturalist*, Vol. 51, No. 3, pp. 422-425.

Shiel, C. B., Shiel, R. E. & Fairley, J. S. (1999). 'Seasonal Changes in the Foraging Behaviour of Leisler's Bats (Nyctalus Leisleri) in Ireland as Revealed by Radio-Telemetry', *Journal of Zoology*, Vol. 249, No. 3, pp. 347-358.

Shirley, P. & Marschner, S. (2009). *Fundamentals of Computer Graphics*, Natic, MA: AK Peters.

Simmons, J. A., Fenton, M. B. & O'Farrell, M. J. (1979). 'Echolocation and Pursuit of Prey by Bats', *Science*, Vol. 203, No. 4375, pp. 16-21.

Songtao, L., Yufei, M. & Wenhui, Y. (2010). 'An Effective Data Fusion and Track Prediction Approach for Multiple Sensors', *Computational Intelligence and Software Engineering (CiSE), 2010 International Conference*, pp. 1-4.

Sun, J., Zhang, W. W., Tang, X. O. & Shum, H. Y. (2005). 'Bi-Directional Tracking Using Trajectory Segment Analysis', *10th IEEE International Conference on Computer Vision (ICCV 2005)*, pp. 717-724.

Surlykke, A., Ghose, K. & Moss, C. F. (2009). 'Acoustic Scanning of Natural Scenes by Echolocation in the Big Brown Bat, Eptesicus Fuscus', *Journal of Experimental Biology*, Vol. 212, No. 7, pp. 1011-1020.

Theriault, D. H., Wu, Z., Hristov, N. I., Swartz, S. M., Breuer, K. S., Kunz, T. H. & Betke, M. (2010). *Reconstruction and Analysis of 3D Trajectories of Brazilian Free-Tailed Bats in Flight*. University of Boston, Computer Science Technical Report.

Tomasi, C., Petrov, S. & Sastry, A. (2003). '3D Tracking = Classification + Interpolation', *Computer Vision Proceedings. Ninth IEEE International Conference*, Vol.2, pp. 1441-1448.

Twente, J. W. (1955). 'Aspects of a Population Study of Cavern-Dwelling Bats', *Journal of Mammalogy*, Vol. 36, pp. 379-390.

Twente, J. W. (1956). 'Ecological Observations on a Colony of Tadarida Mexicana', *Journal of Mammalogy*, Vol. 37, pp. 42-47.

Urbani, C. B. (1980). 'A Statistical Table for the Degree of Coexistence between Two Species', *Oecologia (Berl.)*, Vol. 44, No. 3, pp. 287-289.

Watts, P., Mitchell, E. J. & Swartz, S. M. (2001). 'A Computational Model for Estimating the Mechanics of Horizontal Flapping Flight in Bats: Model Description and Validation', *Journal of Experimental Biology*, Vol. 204, No. 16, pp. 2873-2898.

Whitaker, J. O., Shalmon, B. & Kunz, T. H. (1993). 'Food and Feeding-Habits of Insectivorous Bats from Israel', *Zeitschrift Fur Saugetierkunde-International Journal of Mammalian Biology*, Vol. 59, No. 2, pp. 74-81.

Williams, T. C., Ireland, L. C. & Williams, J. M. (1973). 'High-Altitude Flights of Free-Tailed Bat, Tadarida-Brasiliensis, Observed With Radar', *Journal of Mammalogy*, Vol. 54, No. 4, pp. 807-821.

Witten, I. H. & Frank, E. (2000). *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations*. San Francisco: Morgan Kaufmann.

Yang, J. & Zhang, J. (2007). 'Offline Swimmer Cap Tracking Using Trajectory Interpolation', *Digital Image Computing Techniques and Applications*, *9th Biennial Conference of the Australian Pattern Recognition Society*, pp. 579-585.

Yang, Y. & Chen, K. (2010). 'Unsupervised Learning via Iteratively Constructed Clustering Ensemble', *Neural Networks (IJCNN)*, *The 2010 International Joint Conference*, pp. 1-8.

Zheng, W., Hristov, N. I., Hedrick, T. L., Kunz, T. H. & Betke, M. (2009). 'Tracking a Large Number of Objects from Multiple Views', *Computer Vision*, *2009 IEEE 12th International Conference*, pp. 1546-1553.