Institute for Visualization and Interactive Systems
University of Stuttgart
Universitätsstraße 38
70569 Stuttgart
Germany

Master thesis Nr. 3479

# Optimization of Service Following in Automotive Radio by Applying Service Landscape Memory

Yuan Gao

| | |
|---|---|
| **Course of Study:** | INFOTECH |
| **Examiner:** | Prof. Dr. Albrecht Schmidt |
| **Supervisor:** | Dr. Thomas Eireiner,<br>Prof. Dr. Dieter Fritsch |
| **Commenced:** | 15.11.2012 |
| **Completed:** | 17.05.2013 |
| **CR-Classification:** | I.6.5 |

# Abstract

Digital Audio Broadcasting (DAB) is a method for the digital transmission of radio signals for mobile reception. Compared with the traditional FM and AM broadcasting technology, it has benefits of better audio quality, more services provided and lower transmission cost.

With the promising advantages, DAB technology is widely accepted by various countries for radio broadcasting. A brilliant benefit from DAB is service following, which theoretically provides user a seamless listening experience. With that the radio receiver is able to track a selected service while moving across the boundary of signal coverage.

In this thesis, several ways of improving service following in automotive radio combined with reception prediction are introduced. Therefore algorithms have been developed, simulated with several test drives. Separately the performance of each algorithm is compared with given technical indicators. We analyzed both benefits and drawbacks for each algorithm in order to provide recommendations for the further developments.

Additionally the software, which is used to collect sample data from test drives, is developed in the work.

# Content

# List of Figures

# List of Tables

# 1 Introduction and Background

DAB (Digital Audio Broadcasting), a digital radio technique with the outstanding advantages as CD-level audio quality, higher anti-jamming capability [1], low emitted power, data packet transmission service and more other benefits, is extensive applied worldwide [2]. The advanced development DAB+, which uses the original DAB standard, becomes also increasingly popular which inherits all included benefits from DAB and offers additional functions [3]. Both of them have promising market worldwide.

To overcome the reception coverage issue and improve DAB service, there is a feature - DAB service following, which maintains the same audio or data content the user has selected in spite of varying reception conditions [4], by switching between available radio senders seamless theoretically. Compared with the traditional FM service, there is only service following from FM to FM station provided. However, its bottleneck, which restricts service following's benefit, is asynchronous radio transmission due to different time shift between service providers. There are two solutions to improve the performance; one is by unification of sender and minimizing the time difference, which is caused since each provider has his own way of operation. Secondly, even though in car system, the time shift problem is improved by advanced buffering, it is necessary to minimize the number of switching because they might be very disturbing.

In this thesis we combine the station landscape knowledge[1] with the second solution, to decrease the number of annoying switchings on the premise that DAB audio is provided as much as possible.

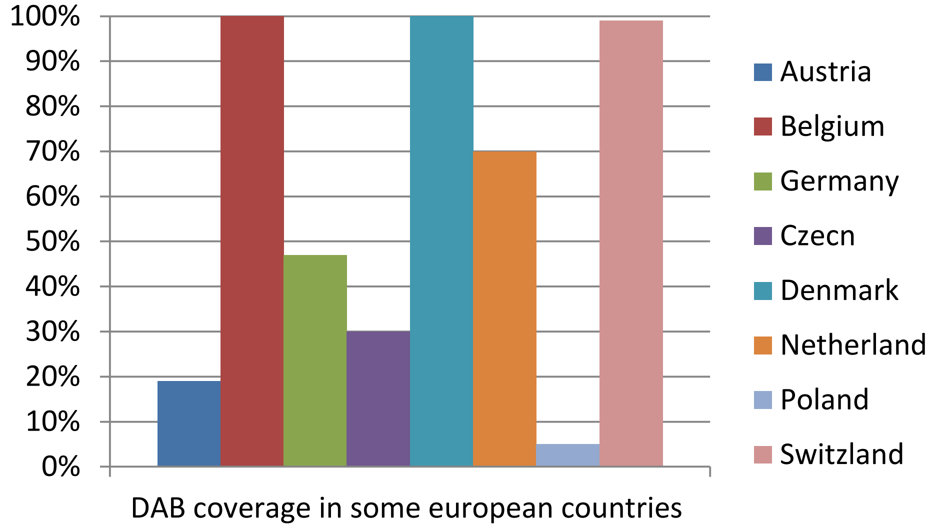
## 1.1 DAB application prospective

The DAB development was first launched in Germany in 1995 in Bavaria [5]. Nowadays it is widely used over the world, especially in Europe. Due to its broader prospective, more countries increasingly devoted themselves into research and development of DAB technology. They are Switzerland, Austria and so on which have satisfactory DAB coverage by decades of effort. Till now the DAB radio has been applied in these pioneer countries substantially as table 1.1 shown.

[1] The knowledge of radio signal reception on the given route

There are also a growing number of developing countries such as Brazil, China and so on joining of the extension of DAB technology.

Table 1.1: DAB coverage in some European countries [6]




As Germany is the first county launching DAB project worldwide, DAB radio is here successfully established and developed. There are various 29 DAB radio ensembles with more than 200 different radio stations in Germany with approximately 47% coverage countrywide [6]. Due to the fact that there are some areas with relative low and uneven broadcasting distribution, service following technology is indispensable by linking DAB and FM. Until DAB coverage has reached an acceptable level in all markets as well as FM radio, whose broadcast area covers nearly 100% of Germany. It allows a seamless listening experience particularly for digital radio on the move consequently.

## 1.2 Service following in DAB tuner

Service information (SI) in DAB is carried in the Fast Information Channel (FIC) as a series of Fast Information Groups (FIGs) carried in Fast Information Blocks (FIBs). Different FIGs are used for different service information and several different FIGs may be needed to implement a particular service information feature such as service linking or announcements. DAB provides signaling that enables broadcasters to inform receivers about the broadcast networks and service configurations that allow service following to take place and also provides dynamic information to control which of that information is used at any given time to take account of changes during the day.

Three types of information may be involved with service following - service linking information, other ensembles information and frequency information. Service following generally provides information to allow precisely the same service to be followed, and when linking information is needed this is called “hard linking”. Broadcasters are also able to indicate suitable alternative content as related services, which are called “soft linking” [7].

There are five main cases applied in service following [4].

1. DAB to DAB link in multi-frequency networks. The same ensemble with the same service is transmitted at different frequency. With the help of synchronization frame [2], it is able to maintain seamless switching.
2. Linking to the same services on different ensembles. The same program service is available on multiple ensembles, and those ensembles are on a variety of frequencies.
3. Linking regional variations of a service in different ensembles. A program service is broadcasted across multiple regions. In this case, the program service does not remain identical all the time within all broadcast regions. By service following it is possible to inform the receivers which services are identical, when the services vary regionally and temporally.
4. Linking technology variations of a service in different ensembles. It is particularly in situation of DAB associated with DAB+, in which the service within different ensembles allows service following even the audio outputs is coded with different technologies.
5. Soft linking of a service. In order to allow receivers to continue provide a service carried on DAB when moving beyond DAB coverage area, service providers with the same audio content available from FM-RDS can provide service following information.

A complete service following is considered as switching among AM (amplitude modulation), FM (frequency modulation) and Web Radio. In this thesis we only concentrate on service following between DAB and FM regardless of cases one to four above. Moreover, our target car system to improve is Mercedes Benz with MOST Catalog NTG4.5 despite of a various number of service following techniques within different automobile manufacturers.

## 1.3 Issues of service following

Since the coverage of DAB service is currently in Germany not overall perfect, some area of the districts have no or discontinuous DAB services as example in figure 1.1[2] [8] below.

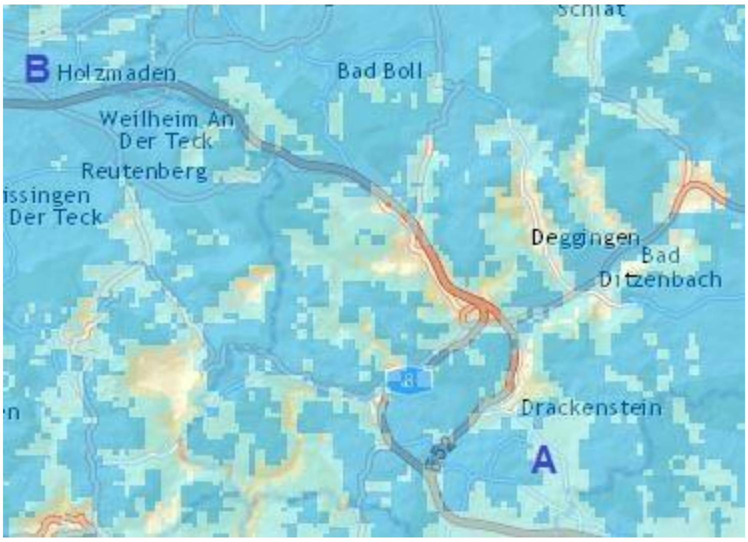



Figure 1.1: Alternative DAB coverage between Ulm and Stuttgart

The typical issue which influences the customer feeling for automotive area is the discontinuous differential listening. Because of the high speed movement of car, user would accidentally jump from one DAB covered area to another. As the issue shown in figure 1.1, the car drives from Ulm to Stuttgart through area between A and B. In order to maintain radio service the user needs to switch to the same radio station in FM radio when crossing the light grey area, which means with no or very weak DAB reception area. Unfortunately the transmitted radio signals from DAB and FM are asynchronous.

User takes serious concerns on fluent broadcast with good audio quality. Unlike a stationary situation, during the driving it might be difficult to remain DAB broadcast due to the incomplete coverage in the real scenario. Especially when driving in area where there is nearly no DAB coverage, the user may hardly receive DAB service. On the other hand, the FM service has much better coverage than DAB service. Service following allows senders to provide listeners seamless experience during the transition between digital and analogue. Radio switches to FM when crossing Non-DAB coverage area, and back to DAB in contrary.

We consistently assume that service following can provide seamless switching to user by conditions that

[2] The color indicates the strength of DAB reception, in which lighter blue means less DAB is receivable

1. Users do not detect audio quality difference between digital and analogue and
2. The broadcast from digital and analogue senders stays strictly synchronized.

The first condition obeys the rule since DAB audio quality is already better than FM. Comparatively a discontinuous audio is indeed more disturbing. Thus the second condition is the key issue to overcome. It may be only optimized by buffering the FM signal inside the car, so that the synchronization is possible.

However for the second attempt, the service following would make the listening experience even worse when crossing the boundaries between DAB covered and Non-DAB area. Fast switching between DAB and FM will lead to a bad user experience, since the user could easier detect different audio quality and delays.

Thus, we try to improve service following by combining the advantages of DAB and FM. This thesis will show, a way how to categorize the disturbances and how to minimize the switchings especially the most annoying ones by maximizing DAB time with best effort at the same time.

# 2 Service Following in Automotive System

After the first attempt in car system, the DAB service has been long term applied and developed in automotive area. The service following plays a significant role for the user experience; therefore the DAB service following within the car is in this chapter introduced. There are many different DAB receiver in the current market. Some of the DAB receiver contain also FM tuner integrated; some especially may cooperate with FM receiver, which are integrated within the Head Unit.

In this chapter, in order to see the implemented service following in the car system, we first introduce the physical implementation of DAB service in our car system together with relevant protocols. Meanwhile as the service following of DAB cooperates with FM service, the functionality of present automotive FM tuner is also brought with. After the general understanding of DAB and FM tuner, the service following from intuitive user viewpoint is introduced.

## 2.1 Radio architecture inside the car

The DAB service provider identity in car is DAB Doppeltuner (figure 2.1). It is via MOST BUS connected in the car system integrated. As figure 2.1 shows, we can via the MOST cable plug connect the tuner on the left side, and connect the car antenna on the right plug. The tuner supports double antenna connection in order to enhance the reception quality and more various services. The actual FM receiver is integrated within the Head Unit in the car. Therefore we cannot provide the outlook of the FM receiver.

The architecture of the radio-related devices connected in the car is shown as figure 2.2. It contains one DAB tuner and FM tuner. FM tuner is currently integrated in the Head Unit. Both of them have double antenna in order to guarantee best signal reception. Through the MOST ring, the amplifier connected with loudspeaker outputs the audio from both of the tuners. Other devices could be TV tuner etc. which serve for the other functionalities.


Figure 2.1: DAB tuner in our car system

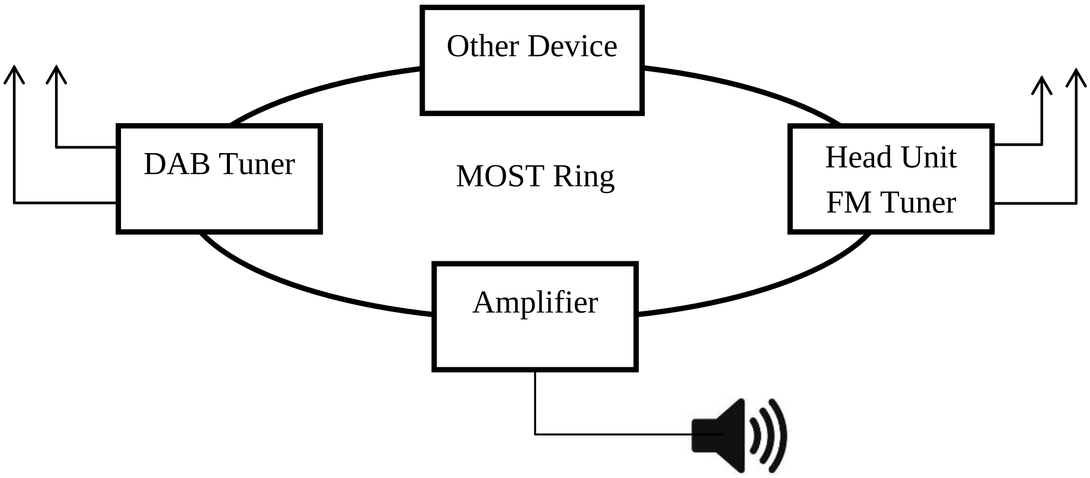



Figure 2.2: Radio architecture inside the car

## 2.2 MOST protocols for DAB/FM tuner

The basic transmission protocol is MOST (Media Oriented System Transport) specification since the communication in the car system is done under MOST connection. Based on the communication standard, the physical identities e.g. DAB tuner and FM tuner are logically abstracted as Function Block [9], in which there summarizes many services from the identities e.g. station name, traffic information. All of these types of logic identity are granted in specific IDs, which are used as the identification data in notification.

The transmission unit via MOST system is called message. A message contains the data as those in typical network protocols e.g. destination address, Function Block ID, Properties etc. Two channels defined in MOST Protocols are Control Channel and Data Channel. Control Channel is responsible for the signaling transmission with relative lower speed; meanwhile Data Channel transfers the data like audio and video with higher speed.

There are two logical components, in which the various tasks are divided by foreground tasks as presenting the current service to listener. Other secondary functions from foreground could be measuring the current signal strength, informing user current station or traffic channel listening. Unlike the foreground tasks, background tasks are for example generation of service lists and lists with alternate frequencies, reception of various data services or background scanning. We call these two logical components as fore- and background tuner.

With the periodic update of new coming techniques, the Function Block of our DAB tuner has several versions. In our experimental test, the version NTG4.5 is used.

## 2.3 Service following in car system

The users achieve multimedia service including DAB and FM radio service. During the drive the user is able to select the services either from DAB or FM. In our car the DAB tuner provides the service following with FM tuner cooperated, which performs a switching between each tuner, see figure 2.3.

The switching behavior is automatically done by DAB tuner. The DAB service provider as the DAB standard agreement defined must integrate the corresponding FM PI code together with the transmitted DAB service. Anytime when the DAB service is not available, the tuner will utilize the given FM PI code to find a suitable DAB service.

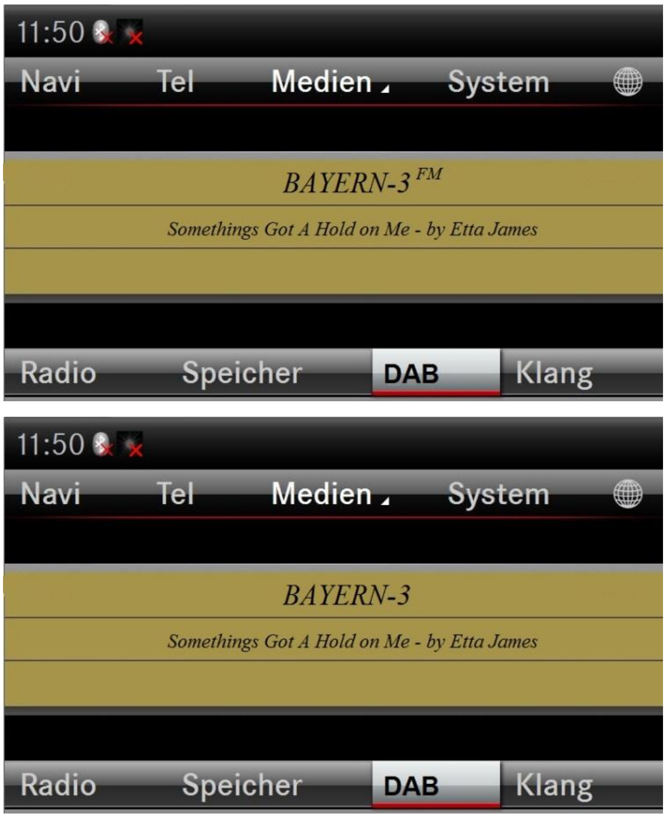



Figure 2.3: DAB tuner switches to FM radio (up) and switches back to DAB (down)

## 2.4 Conclusion

In this chapter, we have introduced the structure of radio system inside our car. The functional components in the systems are DAB and FM tuner, which are presently controlled via the Head Unit. Later we will see, since the tuners can be controlled via the Control Channel in MOST Bus, we can connect extra controller e.g. computer to communicate with each tuner. With sending certain message requiring information of radio, the necessary radio reception information can be achieved.

Since the FM tuner is only controlled by the Head Unit, only one Function Block from FM background tuner is accessible. From the previous discussion in Section 2.2, the information from background tuner in real-time might reply slower than foreground tuner; according to the observation, target function from FM background tuner is quick enough to return message when DAB service following takes place.

We also examine the traditional switching behavior in the current DAB system of the car, which contains already an extra delay before switching back to DAB. It can more or less reduce the fast switching. From our observation, there are a lot of fast switchings, so the chance of performance improvement is promising.

# 3 Sample Data Acquisition

The key requirement of simulation is data sample collection. The sample data should contain radio service information, reception quality parameters, radio ensemble information, latitude and longitude etc. Traditional multimedia system including radio in the car is centralized controlled by the Head Unit, which is the unified hardware interface enabling the communication between user and multimedia system. Since it is hard-coded function, it is difficult for the developer to access. Our team has developed a MOST-USB adaptor, which can be connected in the same MOST rings with other devices. It performs as one communicable identity - computer via MOST-USB adaptor. We can therefore get the information, from DAB and FM tuner, which is available in the MOST Bus.

In this chapter, we set up a grabber program installed in one PC and use the mentioned technique to collect the data for the simulation. The development of program contains hardware and software environment and how to design it. The grabber program should make the data interaction possible between user and device, a user interface is therefore provided.

The implementation of the grabber program is with the help of several existing modules e.g. the MOST-USB adaptor driver and GPS mouse driver. The implementation with these two devices is introduced in this chapter. Relevant knowledge of the data, which are extracted from tuner, is also illustrated for the reader.

This grabber program was installed in our test car system, which were setup to collect data for the simulation. Several test drives were done to collect data for the thesis.

## 3.1 Development environment

The development environment generally speaking contains hardware and software environment. The testing and simulation is done with the test bench which simulate a real environment of driving car in our laboratory. The relevant documentation of grabber program can be found in internal documentation.

### 3.1.1 Hardware environment

The hardware experimental environment for developing the grabber program is a test bench, which is shown in figure 3.1 with relevant components and interfaces. Most of the testing work was done within the development of grabber program before formally installed in the car.

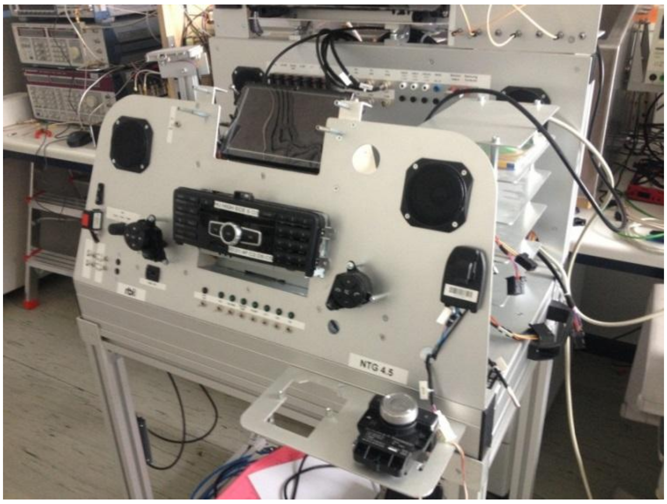



Figure 3.1: Test bench for simulation and testing

In order to examine the data from MOST messages, we use the Optolyzer tool (figure 3.2) with loaded Function Block catalogs. It can constantly monitor all messages within the MOST Bus transported and display them to the developer with agreed format. With its precise and rapid detection of messages, we are able to obtain relevant functions when service following occurs.

Figure 3.2: Optolyzer for MOST message detection

The convertor as a bridge across the PC and MOST ring is MOST-USB Adaptor, which is shown in figure 3.3. With its help the PC is able to connect with the rest of related devices in the MOST ring.


Figure 3.3: MOST-USB adaptor

As the sample radio reception quality is combined with geo-position, the device for capturing GPS signal is required. We use Navilock nl-402 as GPS receiver in figure 3.4. As described in its documentation [10], it is simple to develop via COM port connected with a notebook. 4 Hz update rate is far more enough for the GPS location update in our case [11].

Figure 3.4: GPS receiver Navilock ul-402 for GPS data collection

The hardware environment contains other necessary development tools e.g. test laptop and test car. Table 3.1 below describes shortly the used tools and each usage.

Considered the radio system architecture (figure 2.2), we connected our computer, which installs grabber program to communicate with the tuners, via MOST-USB Adaptor. The new established structure is shown as figure 3.5.

Table 3.1: Tools needed for the development

| **Tool Name** | **Description** | **Note** |
|---|---|---|
| Car with DAB tuner | Used for the test drives | Car type is S204 |
| Lenovo laptop PC | Installed with win32 system | |
| MOST-USB Adaptor | Communication between DAB/FM tuner and PC | Need relevant hardware driver |
| MOST and USB cable | Cable for connection | Cable for connection |
| Navilock NL-402U GPS Mouse | Get GPS data | Parameter setup see Appendix 3 |
| NTG4.5 Test bench | Experimental environment for test and debugging | In the test bench the FM tuner is integrated within |
| MOST Optolyzer | Data via MOST test and debugging | |

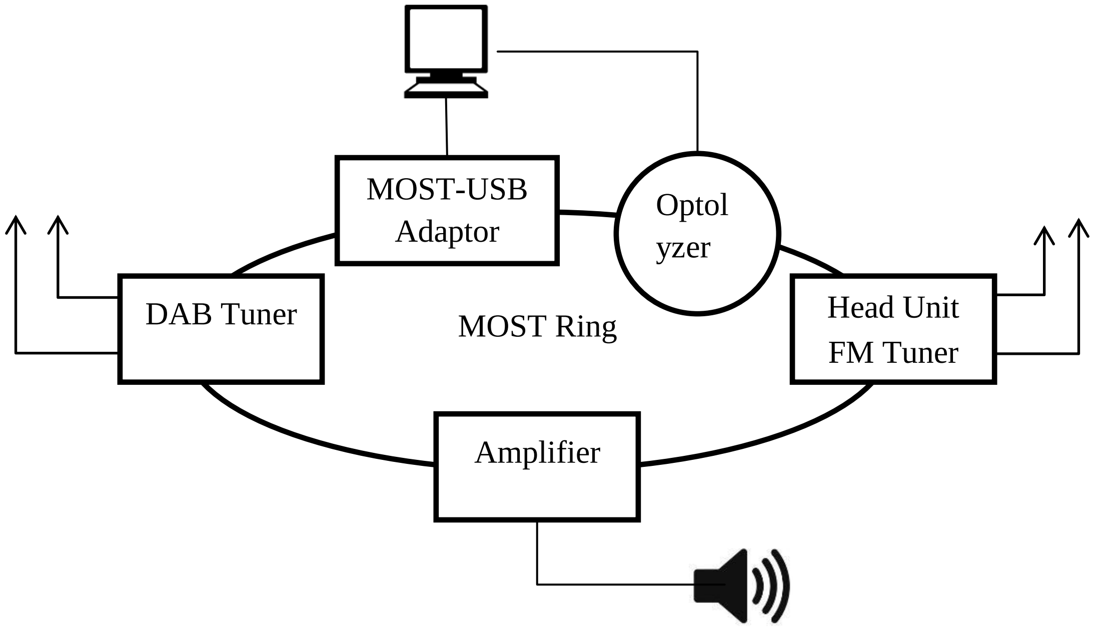



Figure 3.5: Diagram of devices connection

### 3.1.2 Software environment

The software environment consists of system environment and necessary dependencies. Table 3.2 shows briefly the software we used. Till now the development environment is established.

Table 3.2: Software needed for the development

| **Tool Name** | **Description** | **Note** |
|---|---|---|
| QT creator | Development IDE | The relevant Lib is QT 4.8.0 |
| Win32 system | Operating system used | No current release version for Win64 |
| GPS Mouse driver | Hardware driver of the device | Older version is NOT downloadable |
| MOST Optolyzer O4MS V2.1.0 | Optolyzer driver | Do Not use V2.3.0 driver under Win64 |
| Relevant API | Relevant libs for development | See Appendix |

## 3.2 Relevant Sample Data

The sample data to be collected are:

1. Information whether radio signal is receivable or not from tuner.
2. Geo-position information at each sample point.

They are achieved separately by the DAB/FM tuner and GPS receiver.

### 3.2.1 Reception parameter from DAB/FM tuner

Two Function Blocks representing the DAB and FM tuner logically are:

1. **AmFmTuner** (0x40): The functions related to FM tuner [12]
2. **DABTuner** (0x43): The functions related to DAB tuner [12]

As introduced in chapter 2, there are 47 functions implemented in DAB Function Block and 37 for FM tuner. A plenty of flags returned from Function Block are related with radio reception quality. Among them only some bits are of our interest.

Table 3.3: Functions related to DAB/FM reception

| Function Name | Flag Name | Usage of Research | Tuner Type |
|---|---|---|---|
| DTStation_Freq_DC | Frequency | Frequency of the station | FM background |
| | PI | Program index | |
| | Sendername | Station name | |
| CurrentTunerStatus | EnsECC | Extended Country Code | DAB foreground |
| | EnsID | Ensemble ID | |
| | Mute | DAB reception availability.<br>1- DAB not receivable.<br>0- DAB receivable | |
| | SID | Service ID | |
| | SignalQuality | Signal strength. We used Mute flag instead. | |
| AudioServiceList | EnsID | Ensemble ID | DAB back-ground |
| | SID | Service ID | |
| | Label | Ensemble ID name | |

Since the foreground tuner performs quick reaction behavior in signal strength detection, it can reply user data in real-time. We use the data from DAB foreground tuner as input. Meanwhile as the FM foreground tuner currently is not accessible in our car system, which means we only use the FM background information as input. The Function Block returned from FM background tuner is fast enough according to our experience.

To get data from DAB foreground tuner, it is accessible to call CurrentTunerState [12]; for FM foreground tuner, we need to call DTStation_Freq_DC [12]. By the observation from Optolyzer, the listed functions in the table 3.3 are therefore selected for the further research.

### 3.2.2 Geographic information from GPS mouse

The GPS mouse we used provides us the GPS information standardized by NMEA (National Marine Electronics Association) 0183 protocol [13]. According to the standard, there are several GPS strings transmitted containing specific geo-information. In our research only two of them are used to extract the latitude, longitude and altitude data. They are:

1. **GPGGA**: Global Positioning System Fix Data
2. **GPRMC**: Recommended minimum specific GPS/Transit data

The definition of each flag bits of the GPS string can be found in appendix 2.

In the collected sample, it is unavoidable that the reception parameter is not perfect correspondence with geo-coordinate since the geo-location from GPS receiver and radio reception from DAB receiver do not update simultaneously. In practical data sample procedure, we set up the geo-coordinate update period as every 2 seconds. As the distance within 2 seconds' drive is usually very short, we do not consider the deviation in the simulation.

## 3.3 Grabber tool

Depending on the simulation requirement, the grabber program, which is used for data sampling, contains the following functions:

1. Message from DAB tuner retrieving
2. Message from FM tuner retrieving
3. Navigation Information retrieving
4. User Interface with error handling included

### 3.3.1 Software design

The internal logic of grabber program is divided by three parts according to the four requirements in section 3.3.

1. **Data Retriever**: get data from devices like DAB/FM tuner and GPS mouse.
2. **List Process**: output data in an intuitive way e.g. text file.
3. **UI**: contains also user input to indicate car type, weather etc. information.

Three threads were created for each communication with different devices; they are capable to deliver normal control commands and receive data. What's more, the threads are responsible for the data processing in order to extract primary information from the devices. Meanwhile they fulfill error handling to avoid accidental error when data processing fails. The pre-processed data is forwarded to list process thread, which is responsible for storing the pre-processed data with defined format into the text files. UI in this program is responsible for displaying data and accepting input from the user. The logic diagram is shown as figure 3.6.

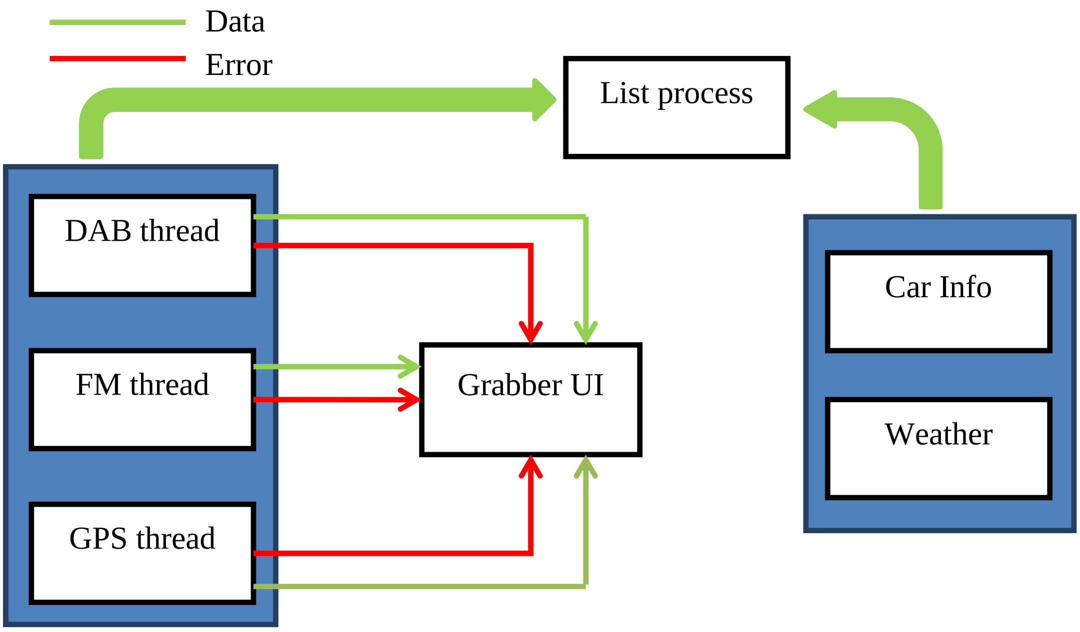



Figure 3.6: Grabber internal logic diagram

### 3.3.2 UI of grabber program

The UI from the grabber program is shown in figure 3.7. The window A shows the available service list from DAB background tuner. Window B shows the available service list from FM background tuner. Additionally the window C respectively shows the information from DAB foreground tuner, which indicates whether the tuner stays in FM or DAB due to service following. Window E is the GPS string which updates itself in every 2 seconds.

Besides the output from devices display, window F is for weather input with three different weather conditions. Window D is the input for car type. The last window on the up-right side is some usage tips for user.

User can also change the directories where the software will store the data. It can do it with given option on the top of the grabber program in “list path”.

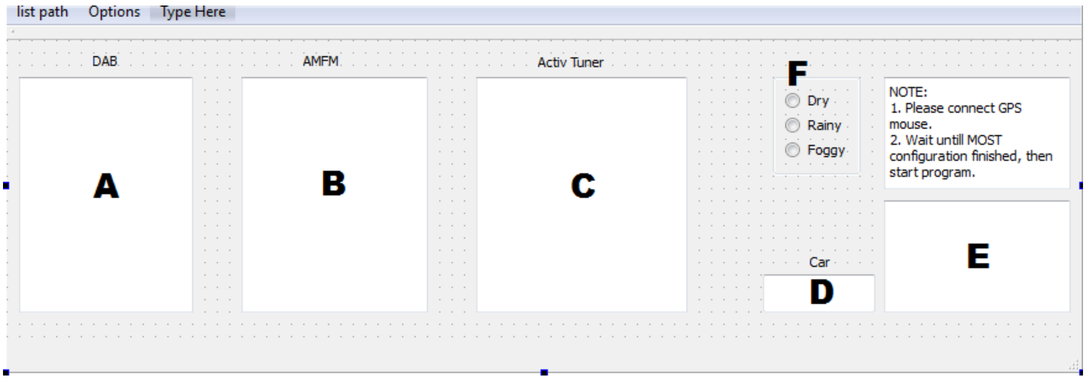



Figure 3.7: UI of grabber program

### 3.3.3 Output of program

The data achieved is stored as text file under Windows environment, which respectively locate in directories name DAB_Foreground, DAB_Background and FM_Background. In the directories, each file name has the same format indicating some information of sample data as: Date_Time[3]_Latitude_Longitude_Level.txt. The example can be found in figure 3.8.

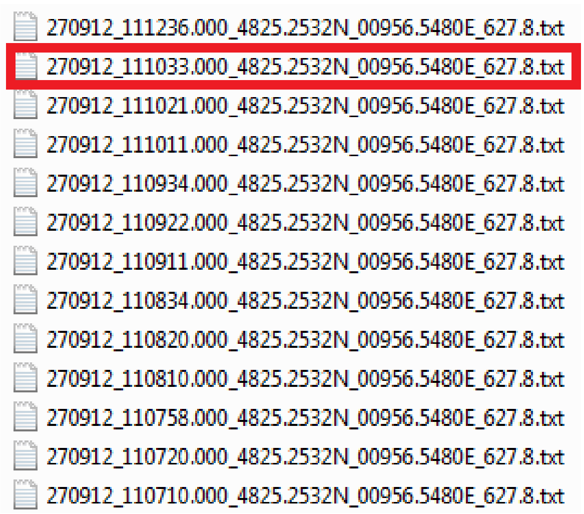
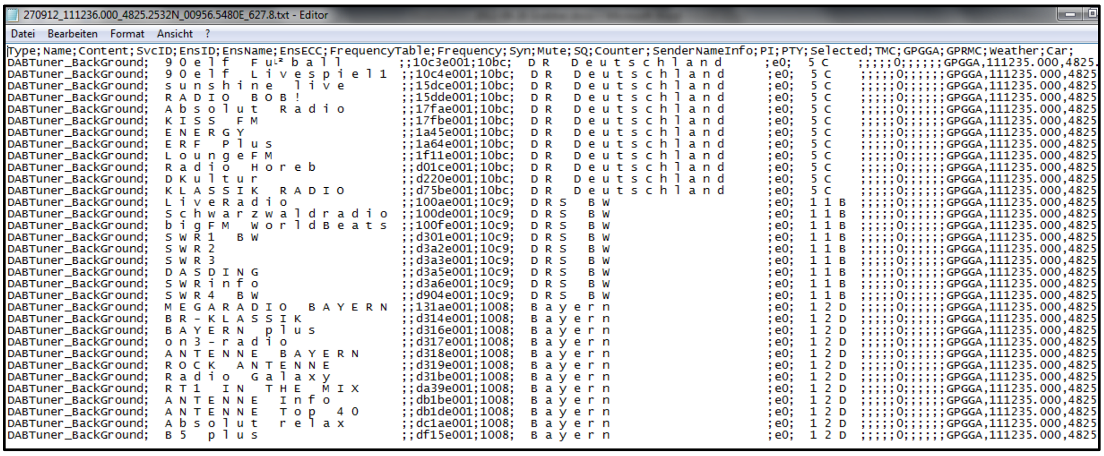



Figure 3.8: Collected files in each directory

Inside the text file there stored the information from each tuner. Whenever a change occurs in the tuner, the tuner via MOST Bus would return the notifications. The notification data is translated and combined with geographic information which is shown in figure 3.9. The meaning of the heading is attached within appendix 1.

[3] The time shown in the list is in coordinated universal time (UTC) format.



```
270912_111236.000_4825.2532N_00956.5480E_627.8.txt - Editor
Datei  Bearbeiten  Format  Ansicht  ?
Type;Name;Content;SvcID;EnsID;EnsName;EnsECC;FrequencyTable;Frequency;Syn;Mute;SQ;Counter;SenderNameInfo;PI;PTY;Selected;TMC;GPGGA;GPRMC;Weather;Car;
DABTuner_BackGround;  9 0 e l f   F u² b a l l        ;;10c3e001;10bc;  D R   D e u t s c h l a n d   ;e0;  5 C    ;;;;;0;;;;;;GPGGA,111235.000,4825.
DABTuner_BackGround;  9 0 e l f   L i v e s p i e l 1 ;;10c4e001;10bc;  D R   D e u t s c h l a n d   ;e0;  5 C    ;;;;;0;;;;;;GPGGA,111235.000,4825
DABTuner_BackGround;  s u n s h i n e   l i v e       ;;15dce001;10bc;  D R   D e u t s c h l a n d   ;e0;  5 C    ;;;;;0;;;;;;GPGGA,111235.000,4825
DABTuner_BackGround;  R A D I O   B O B !             ;;15dde001;10bc;  D R   D e u t s c h l a n d   ;e0;  5 C    ;;;;;0;;;;;;GPGGA,111235.000,4825
DABTuner_BackGround;  A b s o l u t   R a d i o       ;;17fae001;10bc;  D R   D e u t s c h l a n d   ;e0;  5 C    ;;;;;0;;;;;;GPGGA,111235.000,4825
DABTuner_BackGround;  K I S S   F M                   ;;17fbe001;10bc;  D R   D e u t s c h l a n d   ;e0;  5 C    ;;;;;0;;;;;;GPGGA,111235.000,4825
DABTuner_BackGround;  E N E R G Y                     ;;1a45e001;10bc;  D R   D e u t s c h l a n d   ;e0;  5 C    ;;;;;0;;;;;;GPGGA,111235.000,4825
DABTuner_BackGround;  E R F   P l u s                 ;;1a64e001;10bc;  D R   D e u t s c h l a n d   ;e0;  5 C    ;;;;;0;;;;;;GPGGA,111235.000,4825
DABTuner_BackGround;  L o u n g e F M                 ;;1f11e001;10bc;  D R   D e u t s c h l a n d   ;e0;  5 C    ;;;;;0;;;;;;GPGGA,111235.000,4825
DABTuner_BackGround;  R a d i o   H o r e b           ;;d01ce001;10bc;  D R   D e u t s c h l a n d   ;e0;  5 C    ;;;;;0;;;;;;GPGGA,111235.000,4825
DABTuner_BackGround;  D K u l t u r                   ;;d220e001;10bc;  D R   D e u t s c h l a n d   ;e0;  5 C    ;;;;;0;;;;;;GPGGA,111235.000,4825
DABTuner_BackGround;  K L A S S I K   R A D I O       ;;d75be001;10bc;  D R   D e u t s c h l a n d   ;e0;  5 C    ;;;;;0;;;;;;GPGGA,111235.000,4825
DABTuner_BackGround;  L i v e R a d i o               ;;100ae001;10c9;  D R S   B W                   ;e0;  1 1 B  ;;;;;0;;;;;;GPGGA,111235.000,4825
DABTuner_BackGround;  S c h w a r z w a l d r a d i o ;;100de001;10c9;  D R S   B W                   ;e0;  1 1 B  ;;;;;0;;;;;;GPGGA,111235.000,4825
DABTuner_BackGround;  b i g F M   W o r l d B e a t s ;;100fe001;10c9;  D R S   B W                   ;e0;  1 1 B  ;;;;;0;;;;;;GPGGA,111235.000,4825
DABTuner_BackGround;  S W R 1   B W                   ;;d301e001;10c9;  D R S   B W                   ;e0;  1 1 B  ;;;;;0;;;;;;GPGGA,111235.000,4825
DABTuner_BackGround;  S W R 2                         ;;d3a2e001;10c9;  D R S   B W                   ;e0;  1 1 B  ;;;;;0;;;;;;GPGGA,111235.000,4825
DABTuner_BackGround;  S W R 3                         ;;d3a3e001;10c9;  D R S   B W                   ;e0;  1 1 B  ;;;;;0;;;;;;GPGGA,111235.000,4825
DABTuner_BackGround;  D A S D I N G                   ;;d3a5e001;10c9;  D R S   B W                   ;e0;  1 1 B  ;;;;;0;;;;;;GPGGA,111235.000,4825
DABTuner_BackGround;  S W R i n f o                   ;;d3a6e001;10c9;  D R S   B W                   ;e0;  1 1 B  ;;;;;0;;;;;;GPGGA,111235.000,4825
DABTuner_BackGround;  S W R 4   B W                   ;;d904e001;10c9;  D R S   B W                   ;e0;  1 1 B  ;;;;;0;;;;;;GPGGA,111235.000,4825
DABTuner_BackGround;  M E G A R A D I O   B A Y E R N ;;131ae001;1008;  B a y e r n                   ;e0;  1 2 D  ;;;;;0;;;;;;GPGGA,111235.000,4825
DABTuner_BackGround;  B R - K L A S S I K             ;;d314e001;1008;  B a y e r n                   ;e0;  1 2 D  ;;;;;0;;;;;;GPGGA,111235.000,4825
DABTuner_BackGround;  B A Y E R N   p l u s           ;;d316e001;1008;  B a y e r n                   ;e0;  1 2 D  ;;;;;0;;;;;;GPGGA,111235.000,4825
DABTuner_BackGround;  o n 3 - r a d i o               ;;d317e001;1008;  B a y e r n                   ;e0;  1 2 D  ;;;;;0;;;;;;GPGGA,111235.000,4825
DABTuner_BackGround;  A N T E N N E   B A Y E R N     ;;d318e001;1008;  B a y e r n                   ;e0;  1 2 D  ;;;;;0;;;;;;GPGGA,111235.000,4825
DABTuner_BackGround;  R O C K   A N T E N N E         ;;d319e001;1008;  B a y e r n                   ;e0;  1 2 D  ;;;;;0;;;;;;GPGGA,111235.000,4825
DABTuner_BackGround;  R a d i o   G a l a x y         ;;d31be001;1008;  B a y e r n                   ;e0;  1 2 D  ;;;;;0;;;;;;GPGGA,111235.000,4825
DABTuner_BackGround;  R T 1   I N   T H E   M I X     ;;da39e001;1008;  B a y e r n                   ;e0;  1 2 D  ;;;;;0;;;;;;GPGGA,111235.000,4825
DABTuner_BackGround;  A N T E N N E   I n f o         ;;db1be001;1008;  B a y e r n                   ;e0;  1 2 D  ;;;;;0;;;;;;GPGGA,111235.000,4825
DABTuner_BackGround;  A N T E N N E   T o p   4 0     ;;db1de001;1008;  B a y e r n                   ;e0;  1 2 D  ;;;;;0;;;;;;GPGGA,111235.000,4825
DABTuner_BackGround;  A b s o l u t   r e l a x       ;;dc1ae001;1008;  B a y e r n                   ;e0;  1 2 D  ;;;;;0;;;;;;GPGGA,111235.000,4825
DABTuner_BackGround;  B 5   p l u s                   ;;df15e001;1008;  B a y e r n                   ;e0;  1 2 D  ;;;;;0;;;;;;GPGGA,111235.000,4825
```

Figure 3.9: Sample data collected with geographic string

Till now the tool for data sampling is ready. The collection of radio reception and GPS information is further used as the simulation input. The grabber documentation can be found in internal documentation [14].

# 4 Simulation Environment

After getting a lot of sample data from several test drives, we can use the data in further process. Since the sample data is only a set of parameters, it has to be processed in order to suit the simulation environment. Necessary tools and theoretical basis of simulation technique are introduced in this chapter.

The origin of a station landscape database is explained before the simulation process. There we introduce the idea of measurement and possible factors, which may influence the prediction results.

Basically the DAB display time might be reduced since the switching from FM to DAB is avoided. Therefore the simulation is established on the compromise of performance since the more number of switching reduced, the less DAB display time is achieved.

## 4.1 Idea of a station landscape database

The attempt of the research is using the prediction of the station landscape to improve DAB service following; the prediction data therefore is inevitable. After recording of the test drive data, we are able to combine the test drives on a certain route as prediction data.

Station landscape is a term that describes the expected reception quality on a certain area or route. In the thesis, the reception quality means the percentage a station is receivable at a given geo-location. By recording historic experience of the route, we get the knowledge to face the coming unknown situation with these experiences. In order to improve the service following, the knowledge of some certain area is of our interest.

With more experience in the same area, the system becomes more and more familiar with the situation. To get the prediction data, we took several test drives recording the radio reception condition. Since the testing condition changes frequently, it contains many random variables within one drive. By a lot of test drives, the uncertainty can be balanced to general case.

The influences of radio reception could be:

1. Weather. The temperature, rainy or sunny can impact the transmission of radio signal [15].
2. Car type. The antenna characters and performances vary in different car. Some antenna might have worse receivability.
3. Driving direction. In different direction the condition between antenna and base station is not the same.
4. Changing environment. These could be traffic condition, car parking etc. which leads to uncertain reception situations.

Hence, with many test drives, these variables could be generalized so that a prediction data which normally suits to the given route is established.

Considered the practical application, the prediction database should contain two parts. One is located at the server side, which contains the overall data collected from each car in order to provide more precise prediction service. The other stored in the car provides local prediction service, which cooperates with the prediction from server side, for example, pre-filtering the data being updated to the server in order to release the stress of frequently data exchange, local prediction when connection to server is unavailable etc.

With the idea created, the key issue to solve, in this chapter, is to get a valid prediction data.

## 4.2 Software for simulation

Used software in list:

1. Matlab
2. QT Creator

Since Matlab is a strong tool for mathematical and module simulation, we decided the simulation part should be done under Matlab. The demo part should contain a comfortable user-friendly GUI and intuitive visual effect for the user.

## 4.3 Prediction data acquisition

From chapter 3, we have collected many data from each test drive, which are called data samples. Before simulation, these scattered data points must be processed. We first need to merge all the points within one test drive into a list with many points collected in one drive. Then the prediction can be derived from all test drives done in the same routes.

### 4.3.1 Choice of test drive

In order to examine the performance to reduce fast switching, the area which contains many fast switching must be found. According to our experience and practical test, the DAB station “Bayern 3” on the route B312 between Biberach and Memingen has a plenty of discontinuous reception. Therefore, part of the test drives was taken from this area.

On the other hand, we know on most drives the fast switching does not just happen over the whole drive. It is more likely that only within a certain part of the route there is area with plenty of fast switching. Hence, the route between Stuttgart and Ulm, more like general situation, was taken to get sample data.

### 4.3.2 Test data for one test drive

Each tuner itself updates the information independently; therefore we need to combine these data into one data set for each test drive. From table 3.3 we see, there exist many parameters indicating signal reception. In our simulation, we took “Mute” flag in block “CurrentTunerStatus” for DAB. Here 0 means DAB is receivable, 1 means DAB is not receivable. By FM reception it is derived by the frequency with corresponding DAB service in “DT_Station_Freq_DC”. If the frequency exists, then FM service should be receivable, otherwise not. Two of them are used in our merged test drive list.

Furthermore, the update is done asynchronously, which means the data from DAB foreground tuner and FM background tuner is stored respectively in different directories. We can easily merge the data because the route where the data was collected is identical and the record is in time order. With the simple merging, a combined data set is achieved as example in figure 4.1. This example illustrates the idea of merging. Since the DAB tuner only update “Mute” flag, FM tuner only update available station frequencies respectively, when a certain service becomes un-/receivable, the reception between two sample points, should be the same as the earlier point. Specifically in the example in figure 4.1, the DAB reception in 14:58:07 should be the same as the last DAB update in 14:58:02. Therefore, at 14:58:07 the DAB reception should be available.

By using both of the data from DAB and FM tuner, the radio station reception combined with geo-location is prepared. We stored the data in the first two columns as latitude and longitude and with the last two columns as DAB and FM reception. We display the sample data structure (table 4.1)

Table 4.1: Merged input data with geo-location

| Latitude | Longitude | DAB receivable | FM receivable |
|---|---|---|---|
| 48.0649 | 9.9679 | 0 | 1 |
| 48.0649 | 9.9680 | 1 | 1 |
| 48.0650 | 9.9681 | 1 | 1 |
| 48.0649 | 9.9682 | 1 | 1 |
| 48.0650 | 9.9682 | 0 | 1 |
| 48.0650 | 9.9684 | 0 | 1 |
| 48.0651 | 9.9688 | 0 | 1 |

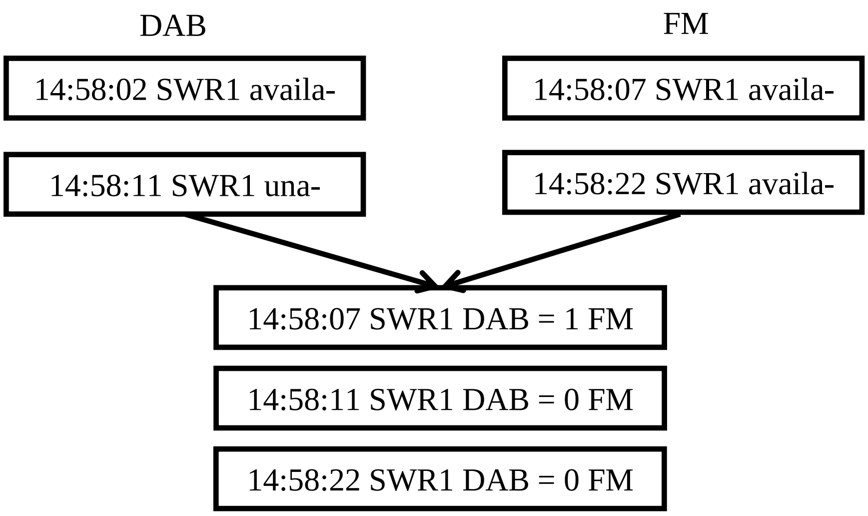



Figure 4.1: Procedures to merge the data from different tuner into one list

### 4.3.3 Prediction derived from several drives

While sample data in different test drive are not geographically ordered, we cannot merge them together with time order.

We take the test route between Biberach and Memingen as example in figure 4.2. The occurrences of green points indicate the DAB is receivable from now on and by contrary the red one means not. After combination, we should get a super set from all the test drives with total number of geographic points. This super set is used by any of the test drive so that each of the test drive can find its prediction point from the super set. One key factor should not be ignored is that the test drive and joint prediction should be separately considered under simulation, e.g. using sample test drive 1 and joint prediction constructed with test drive 2, 3 while not using joint prediction constructed with test drive 1, 2, 3. Each point in the super set indicates the occurrence probability among all the test drives.

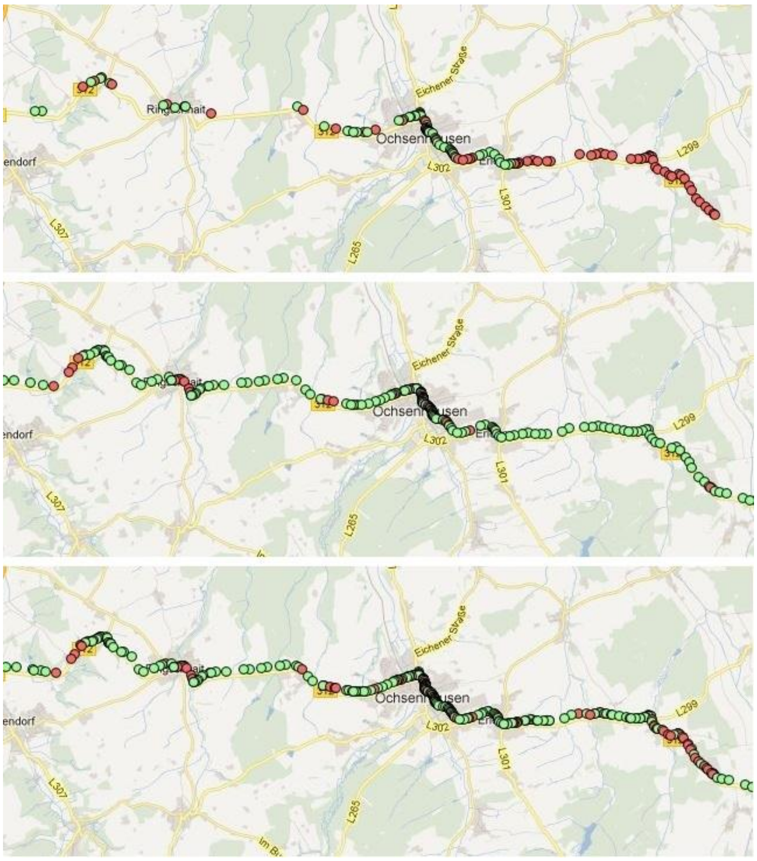



Figure 4.2: test drive 1 (up), test drive 2 (middle) and merged data from them (down)

A typical issue of combining test drives is achieving the correct order of points. As the sample data is recorded unparalleled, the timing stamp of each recorded file is useless to indicate the occurrence order. It is not the same as the technique for merging a complete sample data in one certain test drive, as the sample is ordered by record time stamp for each specific drive.

Therefore the shortest path algorithm to find an optimal way regarding of a bunch of sample data set is required. In our case, the shortest path problem belongs to the Chinese Postman Problem [16], which is described as an optimal route in which the postman traverses all cities only once from the start to destination. In our case, the problem model is not identical as the traditional Chinese Postman since our “postman” should not return.

The classic solution of shortest path is Dijkstra Algorithm, which can absolutely find a shortest path since it traverses all points. However, since the algorithm would traverse all data, with the increasing number of input points, the algorithm becomes far more time consuming [17]. Therefore, we use the method [18] with generic algorithm provided in Matlab, which can reduce much more time to find a nearly optimal path.

The open source code of the shortest path algorithm can be found in the Appendix. Sorting of an optimal shortest path result is shown as the figure 4.3 below. Meanwhile the actual path of the test drive is not identical since in the real scenario the sample data is not with the shortest path principle recorded. Since the data was collected mostly within the drives in highway, the distance from one point to another is almost considered as shortest path.

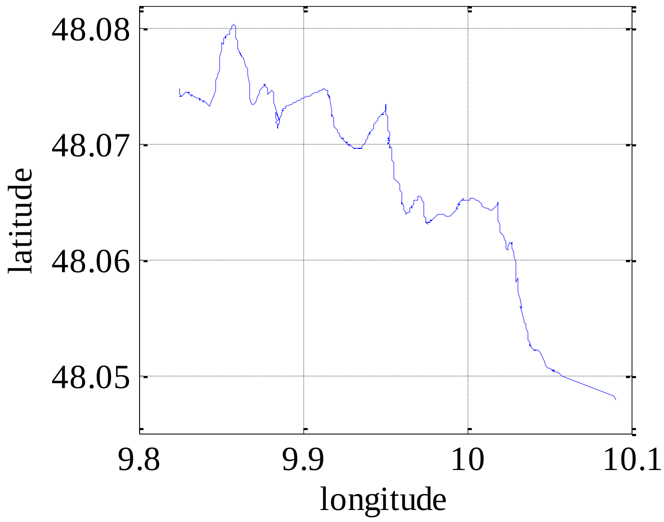



Figure 4.3: Merging several drives using shortest path algorithm from Memingen to Biberach

With the merged sample data set, it is possible to calculate the probability of DAB availability at each point by insertion. The point in a test drive, where no data are available in the geo-location, we took the last available point as the current information. The way we merged data from different test drive is rational. Similar with example in figure 4.1 described, the data between two points should be the same as the last point since update happens in the next point. See example in table 4.2. In this case, the test drive 1 and test drive 3 at the point (45/8) no data, therefore the last reception parameter is taken. "1" means signal receivable and "0" means not receivable. The prediction is therefore calculated by the number of being receivable compared with number of all drives. For example, the value in row second means at point (45/9), we have 75% chance to receive the signal.

Table 4.2: Example how to merge the test drive data into prediction

| **Prediction** | **Latitude/ longitude** | **Test Drive 1** | **Test Drive 2** | **Test Drive 3** | **Test Drive 4** |
|---|---|---|---|---|---|
| 75% | 45/ 9 | 1 | 1 | 0 | 1 |
| 50% | 45/ 8 | X | 1 | X | 0 |
| 50% | 45/ 7 | 1 | X | X | X |

### 4.3.4 Prediction data with time dimension

According to section 4.3 the sample data of a test drive are represented with corresponding geo-location. The simulation should be based on the geo-location from the route start till destination. However, proceeding simulation on geo-position makes the analysis difficult as it is in 2D dimension. Hence, we mapped the data first into 1D dimension based on distance between each point so that the data is marked with distance rather than latitude and longitude. Furthermore, we mapped the data into time by assuming a stable average driving speed which indicates the consumed time from start “0s” to the point.

In this case, the further process would be always based on the given condition that the test car drives constantly in 100 km/h, which turns the first two columns of latitude and longitude from the previous processed sample data into only one column of timing. See example in table 4.3.

With the time-dimension data as input it is easier to simulate the time of DAB display time totally.

In figure 4.4, four of the prediction data are shown, in which the way between Stuttgart and Ulm is derived from 10 test drives and the way between Memingen and Biberach is 6. By comparing the probability of DAB availability we find in some area the DAB reception varies itself frequently which indicates the fast switching area.

Table 4.3: Processed data with time dimension (in second)

| Time | DAB receivable | FM receivable |
|---|---|---|
| 196.8699 | 1 | 1 |
| 198.0284 | 1 | 1 |
| 198.6726 | 1 | 1 |
| 200.3561 | 1 | 1 |
| 201.6469 | 1 | 1 |
| 201.9442 | 1 | 1 |
| 202.2620 | 1 | 1 |

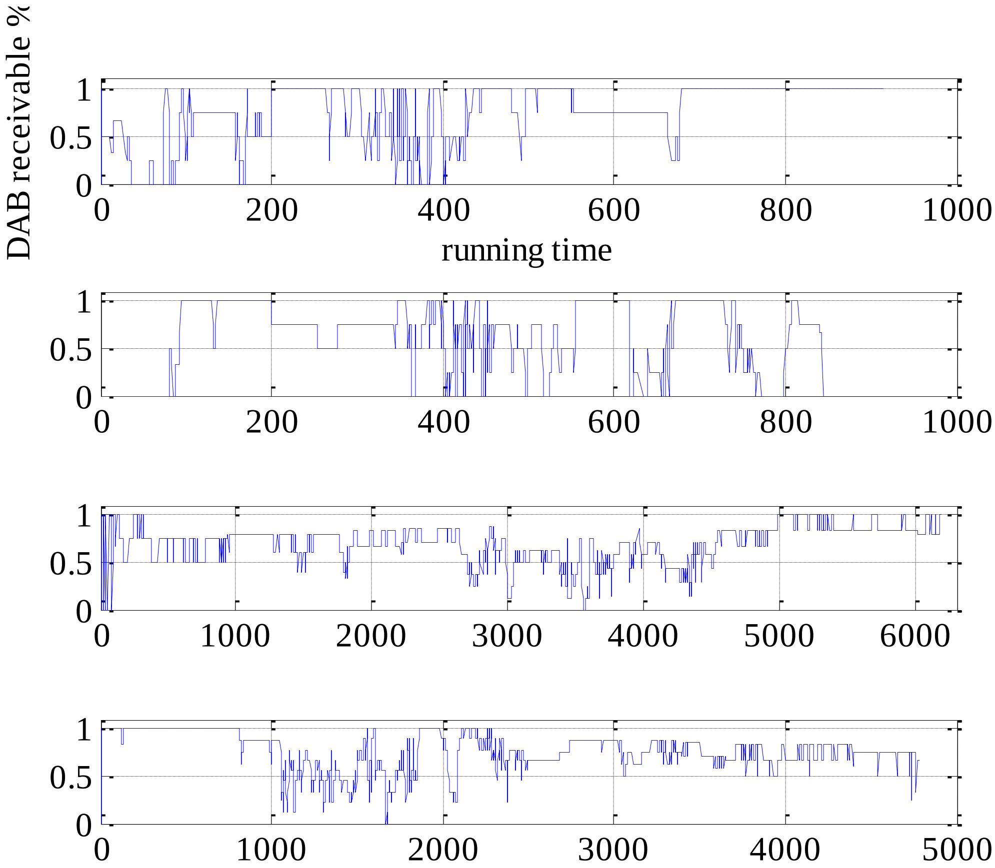



Figure 4.4: Continuous prediction data of from up to down Biberach to Memingen, Memingen to Biberach, Stuttgart to Ulm and Ulm to Stuttgart

According to practical experience, the prediction data should be inversely identical for a certain route. Since the test drives from way backward is not under the same condition e.g. environment, the position of antenna collected, the output of each route back has dissimilarity than its forward way.

## 4.4 Primary algorithm model

The model is created for re-use purpose which fits for the whole simulation. The model itself consists of several inputs and corresponding outputs mapping individually, hence a common way to design a finite state machine is to apply the Moore Machine [19] for the algorithm models. We split the states of models according to real scenarios of user behaviors. Practically a reasonable implementation of real scenario consists of three states:

1. **DAB state**: DAB radio on play, the user is listening to the DAB audio.

2. **FM state**: FM radio on play, the user listening to the FM audio.

3. **No Reception state**: Inaudible/Pure noise representing no radio stations available for the user.

In practical application, the priority of DAB display is higher than FM, when both are receivable, due to the fact that a better radio service DAB should be provided. Actually the No Reception state in implementation is part of FM state and DAB state, in which the user stays in DAB radio with no DAB service receivable or FM radio with noise displayed. To make our simulation process simple, we add this state inside of the model.

The input parameters are from the merged list (table 4.3), where DAB/FM receivable is 1 or 0. Therefore two receptive parameters for each DAB or FM are rational and sufficient:

1. DAB receivable

2. FM receivable

The adequate possibility of different combination of two-bit inputs is four, which covers all possible inputs to our state machine.

Combined with real scenario and each defined states, the output of the model is constructed by three regarding of the theoretical basis that one state has unique corresponding output according to Moore Machine.

1. **DAB on play**. “1” means that the current presented audio is from DAB service and “0” means not.

2. **FM on play**. “1” means that the current presented audio is from FM service and “0” means not.

The two bits can neither be “1” nor “0” meanwhile due to the statuses described overhead, which means it is impossible for the user to listen the DAB and FM radio at the same time. The structure of the state machine is shown as figure 4.5.

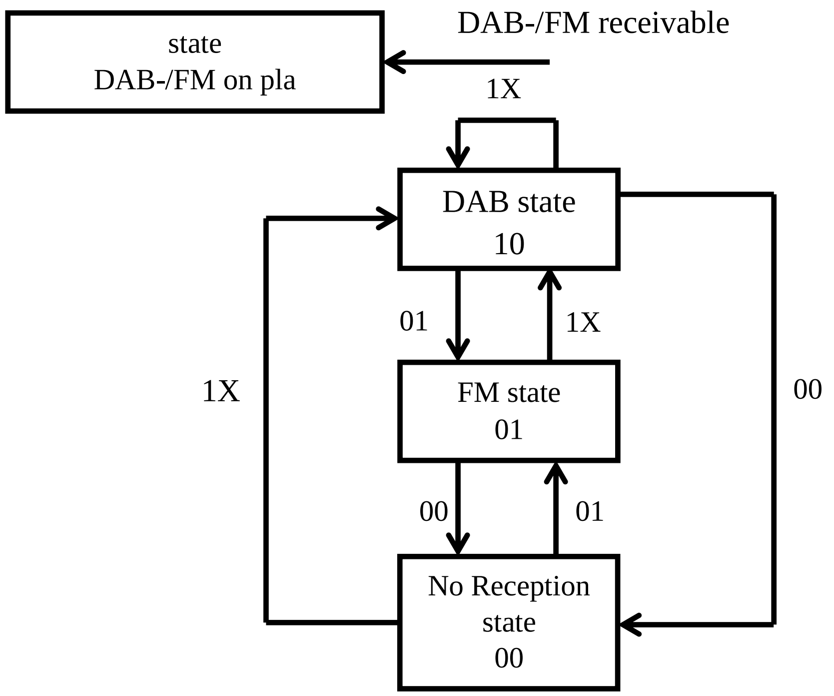



Figure 4.5: State machine of traditional operation

By the table of transition diagram regarding of all possible cases of transition, the validity of state machine is examined. See table 4.4 below.

Table 4.4: Transition of state machine

| Next State / Last State | DAB | FM | No Reception |
|---|---|---|---|
| DAB | 1X | 01 | 00 |
| FM | 1X | 01 | 00 |
| No Reception | 1X | 01 | 00 |

## 4.5 Conclusion

In this chapter, we introduce the necessary prerequisite for the simulation. The necessary simulation inputs are prepared e.g. test drives data and prediction data.

From Chapter 3 it is also known that plentiful parameters provided by DAB tuner indicate the reception quality e.g. “NotReceivedCounter”, “Mute”, “Syn”. In our case, we only used “Mute” flag to indicate the DAB receivable or not.

## 4.5 Conclusion

Due to the existing some seconds delay in the car between DAB to FM switching, the prediction might have deviation which is unavoidable. We cannot change the current running operation in the car, so that the prediction is from our best effort.

Furthermore, the prediction data in real application may be within a discrete tile, which means the prediction value will not be given in continuous way. In our simulation the prediction is assumed to be continuous e.g. sample point interval is usually only within 5 meters. Hence the prediction of the simulation is considered as continuous prediction.

# 5 Improved Algorithms

The previous chapters are only the preparation work before simulation. In this chapter, the improved algorithms compared with traditional behaviors in DAB tuner are solitary illustrated with comparative performances. We will use the pre-defined Moore State Machine to extend each algorithm model in order to speed up the research efficiency.

The algorithms are classified into two types: one with prediction; the other without. Especially for the one with prediction, it is assorted as continuous and discontinuous prediction. In order to clearly compare the different performances, the indicators which are derived from the traditional DAB service following are compared with each algorithm respectively. We cannot provide all of the test data in the paper due to the space limitation, only several typical data therefore are selected to show.

Even in realistic implementation, the prediction is discrete established as described in section 5.8, the algorithm models with continuous prediction is worth discussing, in order to obtain the develop prospective.

## 5.1 Performance indicators

As it is illustrated in chapter 1, one of the major drawbacks of service following is asynchronous audio switching, caused by instable DAB reception condition, particularly in intersection areas. Obviously this defect is paradoxical against service following quick and precise reaction principle. From the previous chapters we know there are two ways to improve the customer feeling intuitive,

1. **Audio quality**
2. **Radio content synchronization**

The first factor is not taken into account to our suggestion as it can be only solved by improve FM audio quality to the similar level as DAB audio. In addition, the traditional way to reduce the existing delay is by buffering audio in advance. Still in any case, to reduce the number of fast switching is one of the important tasks in improvement.

Before the comparison of each performance, a set of standard performance indicators should be defined. The performance indicators should be also based on the customers' demands technically.

The performance indicators encompass:

1. **Percentage of DAB display time along the road**
2. **Total number of switching from FM to DAB audio and vice versa**
3. **Switching interval distribution (<5s, <30s, <180s, <600s)**[4]

Other performance indicators may also be interesting. In the thesis, we only take the listed 3 indicators as the only criteria the compare the following algorithm models.

From the side of customers, a seamless audio experience requires as less as possible Service Following between DAB and FM. On the other hand, he would like to hear audio sound with higher quality from DAB services. Hence the performance indicators associated with customer-oriented consideration are based on the principle as follow:

1. **remain as much as possible DAB audio time**
2. **reduce as many as possible number of switching between DAB and FM**
3. **reduce as many as possible number of Fast switching (switching interval less than 5s and 30s) between DAB and FM**

The performance indicators are used to judge each algorithm later. Particularly the switching interval less than 5s and 30s is only example taken as simulation factors. They are considered in our work as the most disturbing fast switchings. On the other hand, the relative fast switching is meanwhile disturbing e.g. 150s or 180s; since they are secondary important compared with the other two, we do not regard them as the key conditions to improve.

## 5.2 Traditional algorithm performance

Based on the standard description of service following technique, the switching behavior which runs in the car system, contains no intentional delay. In other word, the DAB tuner performs purely immediate switching, in which the tuner switches between FM and DAB radio when one of the others' reception levels sinks under the threshold, according to detected reception levels. Certainly, the listening audio becomes unavailable, when both of the signal strengths are too low to fulfill limit of the antenna in the car.

According to the practical experience with the current DAB tuner, the switching from FM to DAB is not as quick as possible done. We see the DAB tuner in the car fulfills already certain length of delay to avoid disturbing switching. Comparatively the inten-

[4] The number of the switching in different intervals simulated from 0 to given time value.

tional delay is still not satisfactory based on the customer feeling. Therefore in the further extension the existing delay would be enlarged. Due to the primary assumption, we consider that the implementation maximizes the display time of DAB audio on the whole drive, since it reaches the obtainable DAB reception with best effort regardless of number of switching.

Along the data sampling procedure, four routes as typical scenarios of Service Following operation are chosen. Each of them is repeated to record 5 to 15 times. In our simulation only three of the test drives from each route are selected to compare the performance. As agreed performance indicators described in section 5.1, the original performance in current DAB tuner is shown in table 5.1 below.

Table 5.1: Performance of traditional algorithms

| Test Drive[5] | DAB Time[6] | Total Switch[7] | Switch[8] < 5s | Switch < 30s | Switch < 180s | Switch < 600s |
|---|---|---|---|---|---|---|
| 2012-08-15 S to U | 93.55% | 27.422 | 7.94 | 19.48 | 23.81 | 24.26 |
| 2013-01-21 S to U | 49.89% | 49.56 | 10.84 | 32.52 | 43.36 | 48.01 |
| 2013-02-19 S to U | 95.62% | 20.08 | 5.32 | 12.40 | 17.13 | 19.49 |
| 2012-08-31 U to S | 82.63% | 62.97 | 22.37 | 49.72 | 59.66 | 61.32 |
| 2013-02-07 U to S | 62.32% | 90.28 | 32.24 | 70.93 | 87.05 | 88.13 |
| 2013-02-15 U to S | 86.03% | 43.89 | 5.98 | 27.93 | 41.89 | 41.89 |
| 2013-03-06 B to M 1 | 87.92% | 204.02 | 132.02 | 184.02 | 200.02 | 200.02 |
| 2013-03-06 B to M 2 | 89.40% | 198.44 | 141.74 | 186.29 | 194.39 | 194.39 |
| 2013-03-15 B to M 3 | 76.03% | 318.94 | 196.55 | 286.85 | 313.41 | 313.41 |
| 2013-03-06 M to B 1 | 45.96% | 304.50 | 170.95 | 272.45 | 299.16 | 299.16 |
| 2013-03-06 M to B 2 | 83.27% | 308.01 | 199.02 | 274.84 | 298.53 | 303.27 |
| 2013-03-15 M to B 3 | 62.00% | 291.67 | 174.06 | 254.04 | 286.97 | 286.97 |

From the table we see in the route from Memingen to Biberach contain lots of fast switching, which reach approximately 66% of the whole number of switching. In average every 20s to 30s, occurs a fast switching, which is smaller than 5s. Obviously the fast switchings happen very frequently, so that the original performance is dissatisfactory for the user.

[5] Name of each test drive recorded. S-Stuttgart, U-Ulm, B-Biberach and M-Memingen.
[6] Percentage of DAB display time on the whole drives
[7] Number of FM to DAB radio switching per hour
[8] Number of FM to DAB and DAB to FM switching per hour, which are smaller than given time.

## 5.3 Algorithm with longer hysteresis

The section 4.2 illustrates the primary performance of DAB service following. From the output results we discover an interesting point that the switching between FM and DAB relatively centralized in certain areas rather than scattered distribution, which means the fast switchings gather intensively in certain area.

It is consistent with the fact that the DAB coverage is determined by the base stations where they are located, regardless of other environmental effects. With the further distance away from base station, the weaker signal we receive. Eventually in the intersection area the reception becomes unstable due to the more complicated interference factors where the switchings mostly happen.

In this algorithm, the handover procedure is made intentionally delays $t_D$ in order to avoid high frequently receptive varying. In other words, the slower reaction of handover ignores the switching within reaction time. This might come to the case that it even reduces the DAB display time as a compromise of decreasing the number of switching.

The original Moore Machine model applied in last chapter has been re-implemented.

### 5.3.1 Algorithm model

The basic idea is to implement a counter to add a certain delay in order to avoid fast switching. It provides the DAB receiver a “waiting” function.

In order to avoid the fast switching from FM to DAB, the “waiting” function with longer hysteresis can reduce switching within this hysteresis obviously. This is the primary idea why longer hysteresis is introduced.

Consider still the model we applied in previous work and also with the analysis above, we add one more state – “Waiting state” to distinguish the DAB state in the previous section (figure 4.5). The DAB state in new model remains the same as the before, the new coming state considers the case regarding of counter. The state machine is defined as in figure 5.1.

1. **DAB state**, DAB radio on play, the user is listening to the DAB audio.
2. **FM state**, FM radio on play, the user listening to the FM audio.
3. **No Reception state**, Inaudible/Pure noise representing no radio stations available for the user.
4. **Waiting state**, FM is still displayed.

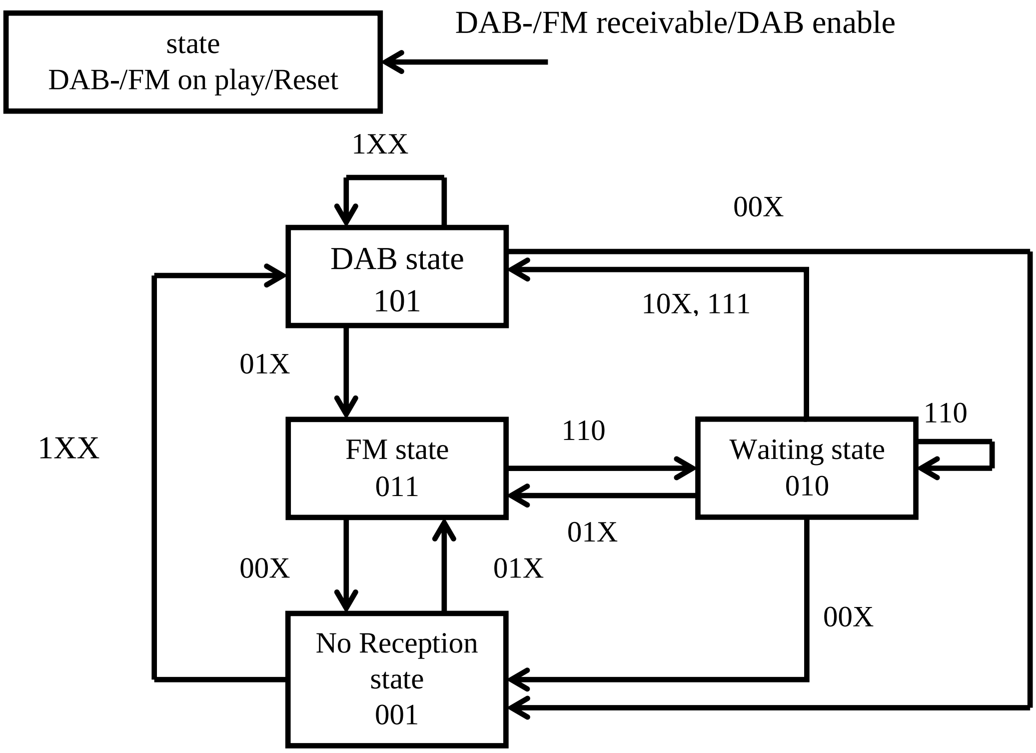



Figure 5.1: State machine with fixed delay

The Waiting state is derived from counter mentioned above. Any time when DAB signal is able to receive from FM state, the machine enters the Waiting state with given waiting time. The length of waiting time is represented as a counter, which could be zero or some short seconds depending on the algorithm itself. Consequently the user hear only FM radio before the counter times up even when DAB radio is available.

We take the input parameter as applied in previous work. A new input, which is the output from timer logic, is added. It covers all the possibilities with totally three input parameters. This extra input is considered as the input to indicate when FM should switch to DAB. The three inputs are as followed:

1. **DAB receivable**
2. **FM receivable**
3. **DAB audio enable**

They cooperatively determine the state transition in the state machine, see table 5.2. The table describes all possible transitions in the state machine. For example, the value “1XX” in column 2 and row 2 means DAB signal is receivable regardless of the rest of two bits. This leads to the transition from DAB to DAB state, which in reality indicates the user stays in DAB radio listening when DAB signal remains receivable.

Table 5.2: Transition of the state machine

| Next State / Last State | DAB | FM | No Reception | Waiting |
|---|---|---|---|---|
| DAB | 1XX | 01X | 00X | - |
| FM | - | 01X | 00X | 110 |
| No Reception | 1XX | 01X | 00X | - |
| Waiting | 111, 10X | 01X | 00X | 110 |

Combined with real scenario and each defined states, the output of the model is constructed by three regarding of the theoretical basis that one state has unique corresponding output according to Moore Machine. Again, the first two bits can neither be “1” or “0” at the same time.

1. **DAB on play**. “1” means that the current presented audio is from DAB service and “0” means not.

2. **FM on play**. “1” means that the current presented audio is from FM service and “0” means not.

3. **Reset**. “1” means timer is reset and “0” means not.

Before discussing the added bit “Reset”, we drift deeper into the implementation of DAB enable logic shown as figure 5.2.

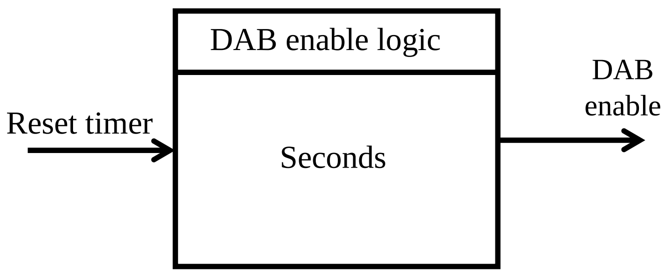



Figure 5.2: DAB enable logic of new algorithm with longer hysteresis

A typical DAB enable logic contains the reset input and trigger output to indicate the state machine transit from Waiting to DAB state. In our DAB enable logic the output is a trigger bit to enable DAB display. The length of the waiting time inside the enable logic is a parameter stored in the logic.

Since the DAB enable logic is attached to the Waiting state, which causes intentionally switching delay until waiting time ends, the "Reset" flag as the last bit of output would enter enable logic to indicate whether the counter should be reset. The complete logic is shown figure 5.1.

If delay length is set to zero, the model will be the same as the traditional algorithm.

### 5.3.2 Simulation result

Since we introduce delay at each service following from FM to DAB, the frequently switching inside the intersection area is outstandingly decreased. We use a delay 30s as example since the delay time is not yet fixed. 30s is definitely longer than the implemented delay in the current DAB tuner. To find an optimal delay length is also of our interest, which is discussed in section 5.7. The performance of algorithm is shown in table 5.3.

Table 5.3: Performance of algorithm with longer delay with delay 30s

| Test Drive | DAB Time | Total Switch | Switch < 5s | Switch < 30s | Switch < 180s | Switch < 600s |
|---|---|---|---|---|---|---|
| 2012-08-15 S to U | 86.60% | 12.99 | 0.00 | 0.00 | 9,38 | 10.82 |
| 2013-01-21 S to U | 37.71% | 21.68 | 3.10 | 4.65 | 17.04 | 20.13 |
| 2013-02-19 S to U | 89.75% | 11.81 | 0.59 | 1.77 | 9.45 | 11.22 |
| 2012-08-31 U to S | 70.21% | 14.91 | 0.00 | 0.00 | 9.94 | 12.43 |
| 2013-02-07 U to S | 43.63% | 26.87 | 0.00 | 6.45 | 23.64 | 24.72 |
| 2013-02-15 U to S | 73.43% | 22.94 | 1.00 | 5.98 | 20.91 | 20.95 |
| 2013-03-06 B to M 1 | 59.05% | 36.00 | 0.00 | 0.00 | 32.00 | 32.00 |
| 2013-03-06 B to M 2 | 70.57% | 28.34 | 0.00 | 4.05 | 24.30 | 24.30 |
| 2013-03-15 B to M 1 | 39.67% | 53.12 | 0.00 | 15.94 | 42.59 | 47.81 |
| 2013-03-06 M to B 1 | 8.92% | 32.05 | 5.34 | 10.68 | 21.36 | 26.71 |
| 2013-03-06 M to B 2 | 41.79% | 61.60 | 0.00 | 18.95 | 56.12 | 56.86 |
| 2013-03-15 M to B 2 | 27.14% | 56.45 | 9.41 | 14.11 | 51.75 | 51.75 |

Considered the route Memingen to Biberach in table 5.3, it is clear that the algorithm with 30s delay reduces much number of the fast switching (less than 5s and 30s) compared with the traditional algorithm (table 5.1). In the original algorithm, there are approximately 180 fast switchings per hour, which are now nearly dismissed. However, the DAB display time compared with previous performance has up to 50% percent DAB time loss. With the growth of delay, the compromise of losing DAB display time becomes unworthy since nearly no DAB time is provided e.g. delay larger than 70s in figure 5.3.

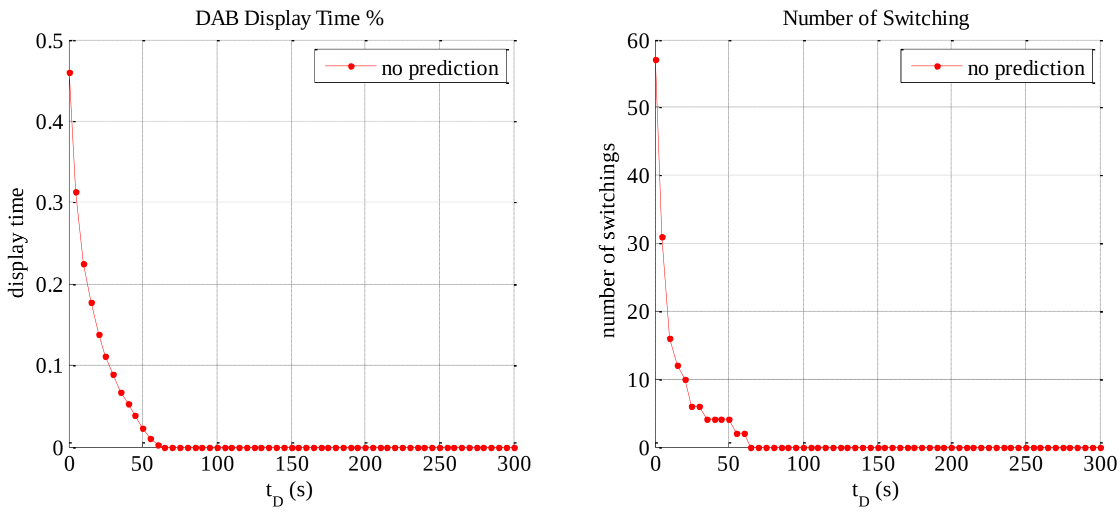



Figure 5.3: Performance of algorithm in route 2013-03-06 Memingen to Biberach

By a given condition that DAB time is 5% and 10% loss, the number of fast switching has been greatly reduced. Especially with DAB time 10% decreased, the number of fast switching (<5s) is half of the original already, shown in figure 5.4[9]. Even with only 5% DAB time decreased, the number of fast switching (<5s) is 30% reduced compared with the original algorithm.

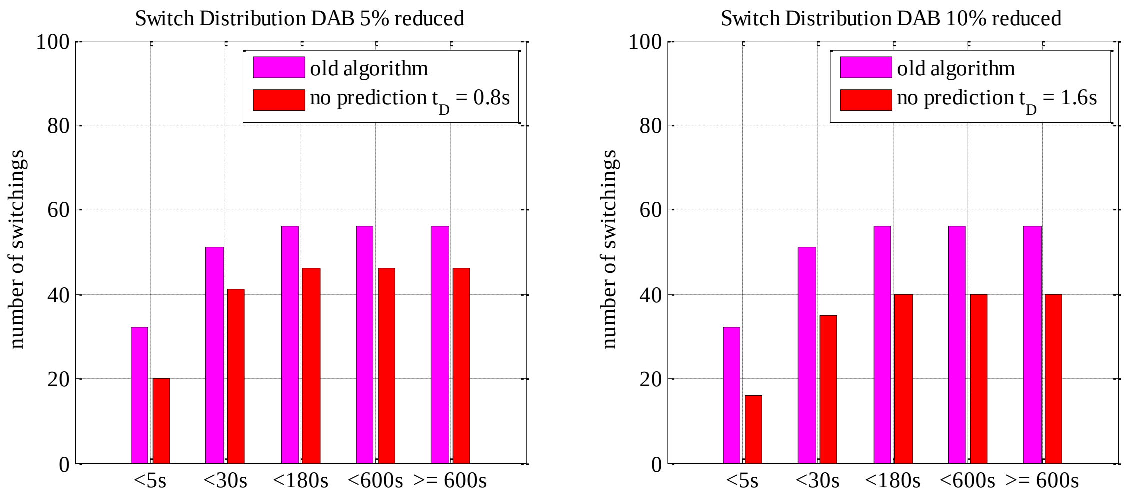



Figure 5.4: Fast switching reduction compared with old algorithm in route 2013-03-06 Memingen to Biberach

[9] Since the reduction of 5% and 10% may be bearable for the user, in the rest of performance comparison, the condition when the DAB display time is reduced by 5% and 10% is always used as example.

## 5.4 Algorithm with prediction in fixed prediction threshold

From the previous section we see the performance between the traditional algorithm with quick switching and the improved algorithm with longer delay switching. By letting the tuner wait for a certain time as delay, the number of switching is dramatically reduced, in which the fast switching part. However, a significant drawback of the algorithm in section 5.3 makes the situation into dilemma.

It is clear that with the increasing delay time, the DAB display time would also decrease. Some DAB display time decrease is meaningful because the DAB signal in fast switching area is ignored, some decrease is however unnecessary because the delay time also makes the DAB signal in slow switching area ignored. As the tuner itself has no knowledge of the coming future, a fixed delay is introduced anytime when DAB signal becomes receivable again. This might reduce DAB display time without any number of switching decrease. If there is a way which can provide the tuner knowledge, to indicate where the fast and slow switching area exists, the unnecessary DAB time sacrifice may be avoided.

With the use of prediction data, we can achieve the knowledge of the radio signal reception quality of given geographic points, which provides the knowledge of the reception distribution for further development. In real implementation, the route is divided by certain area tiles, in which one discrete prediction data is allocated. In order to simplify the discussion, the early attempt would be made under continuous prediction and simulated to get the performance comparison. The discrete case would be discussed in section 5.8.

Since the prediction data represents the probability of receiving DAB, e.g. 60% of the cars can receive the DAB service at that point, the value cannot directly be applied in the algorithm. The easiest way of quantization is by setting up a fixed threshold. The first attempt is to use a permanent threshold to simulate test drives. One example for the algorithm is explained in figure 5.5. The prediction represents the probability of receiving DAB, in which the height of the point means the higher possibility. By setting up a fixed threshold, the prediction whether DAB is receivable or not within the look-ahead time[10] $t_{LA}$ is determined. The user get a feedback that there exists a point, where the DAB service may be not receivable (the third point). Therefore, the user will stay in FM service within look-ahead time $t_{LA}$. In this figure, at the time where the user asks for prediction, the real DAB reception in the future actually is unknown. Therefore we mark the point in measured data as dotted line. We also noticed that if the threshold is set lower (lower than the probability of the third point), the determined prediction would change.

[10] This is the desired time period the user looks into the prediction in the future.

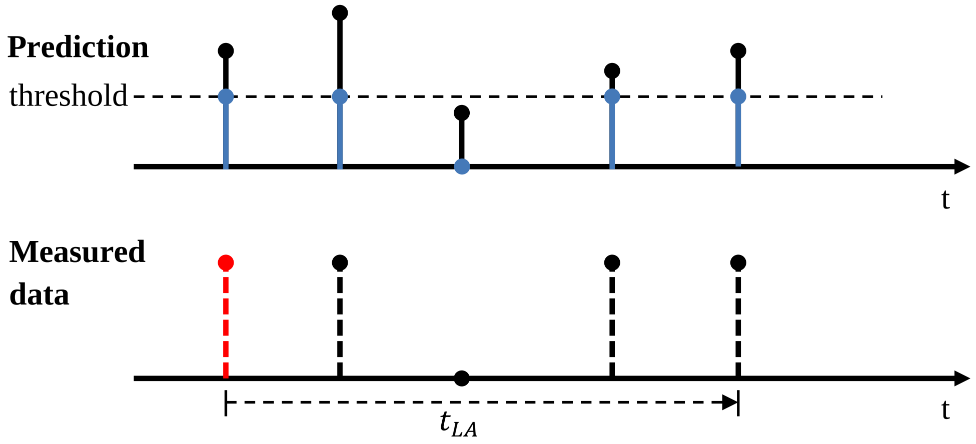



Figure 5.5: Explanation of algorithm with fixed threshold[11]

### 5.4.1 Algorithm model

From section 5.3, we extended the state machine with 4 states and relevant inputs/outputs. The Waiting state applies a DAB enable logic with fixed delay. From the idea of the algorithm itself, the logic should have knowledge of the future in order to determine whether stays in FM or switches to DAB. Therefore instead of the "Reset" flag, we replace the input as "Prediction". Hence the logic itself has the knowledge of future, which leads to the intelligent DAB enable logic determining the output "DAB enable" independently. See figure 5.6 below.

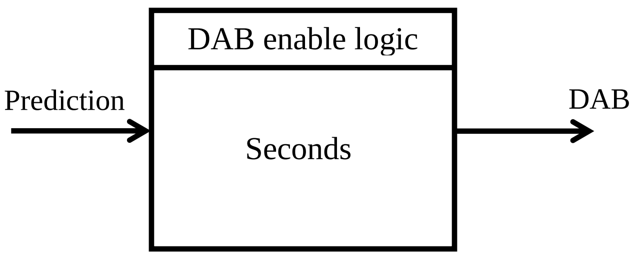



Figure 5.6: DAB enable logic with prediction

In the new DAB enable logic (figure 5.7), the time is no longer delay $t_D$ as in section 5.3. Instead, the time is look-ahead time $t_{LA}$. The logic "Thresholding" uses the predefined fixed threshold to quantize the prediction. According to the probability, if the

[11] Measured data is the real time data which is measured by the car during test drive.

probability is higher than threshold, the prediction data at that point is set to 1, otherwise is set to 0.

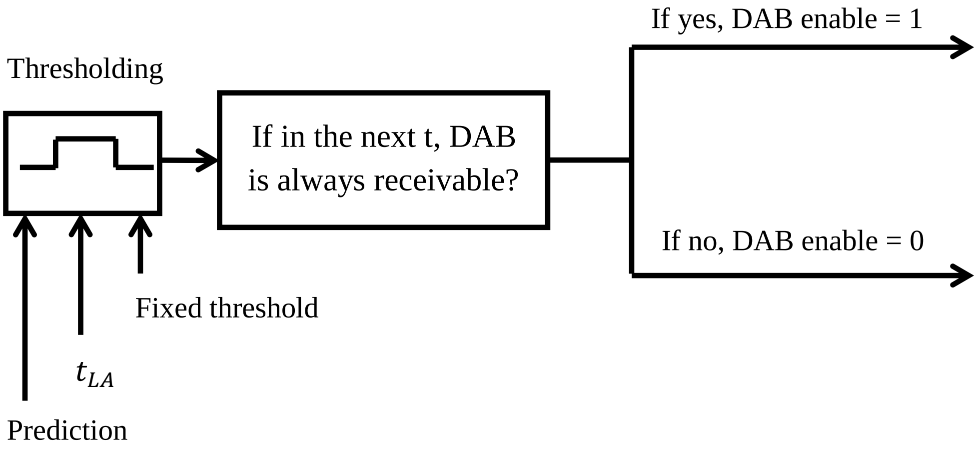



Figure 5.7: Internal logic of new DAB enable logic

The benefit of the model is avoiding big structure change of the state machine model, so that we can re-use the model. The small difference for input/output in the new model is that the bit “Reset” is dismissed.

### 5.4.2 Simulation result

We take a threshold 0.5 to see the performance with delay 30s in table 5.4.

Table 5.4: Performance of algorithm with fixed threshold 0.5 with delay 30s

| Test Drive | DAB Time | Total Switch | Switch < 5s | Switch < 30s | Switch < 180s | Switch < 600s |
|---|---|---|---|---|---|---|
| 2012-08-15 S to U | 86.60% | 12.25 | 2.16 | 4.33 | 7.93 | 10.10 |
| 2013-01-21 S to U | 38.42% | 21.68 | 3.10 | 12.39 | 15.48 | 20.13 |
| 2013-02-19 S to U | 94.29% | 10.04 | 0.59 | 2.95 | 7.09 | 9.45 |
| 2012-08-31 U to S | 80.18% | 21.54 | 0.83 | 7.45 | 16.57 | 19.88 |
| 2013-02-07 U to S | 54.64% | 31.17 | 3.22 | 9.67 | 27.94 | 29.01 |
| 2013-02-15 U to S | 81.47% | 16.96 | 0.00 | 4.99 | 11.97 | 14.96 |
| 2013-03-06 B to M 1 | 78.22% | 64.00 | 20.40 | 40.00 | 50.99 | 50.99 |
| 2013-03-06 B to M 2 | 83.09% | 60.74 | 32.34 | 40.54 | 56.76 | 56.76 |
| 2013-03-15 B to M 1 | 30.15% | 79.68 | 47.79 | 58.41 | 74.34 | 74.34 |
| 2013-03-06 M to B 1 | 9.21% | 21.37 | 5.34 | 10.68 | 16.02 | 16.02 |
| 2013-03-06 M to B 2 | 45.60% | 42.65 | 4.74 | 14.21 | 33.16 | 37.89 |
| 2013-03-15 M to B 2 | 36.58% | 42.34 | 9.41 | 18.82 | 32.94 | 37.65 |

Compared with the previous algorithm without prediction, the reduction of number of switching with fixed prediction threshold can also reduce fast switching. However, in some cases this algorithm performs worse than the algorithm in section 5.3 (see table 5.3), because it causes nearly the same DAB time loss, but many fast switching are still left e.g. 2013-03-06 B to M 2 and 2013-03-15 B to M 1. Two example performances with different fixed threshold[12] are shown in figure 5.8 below.

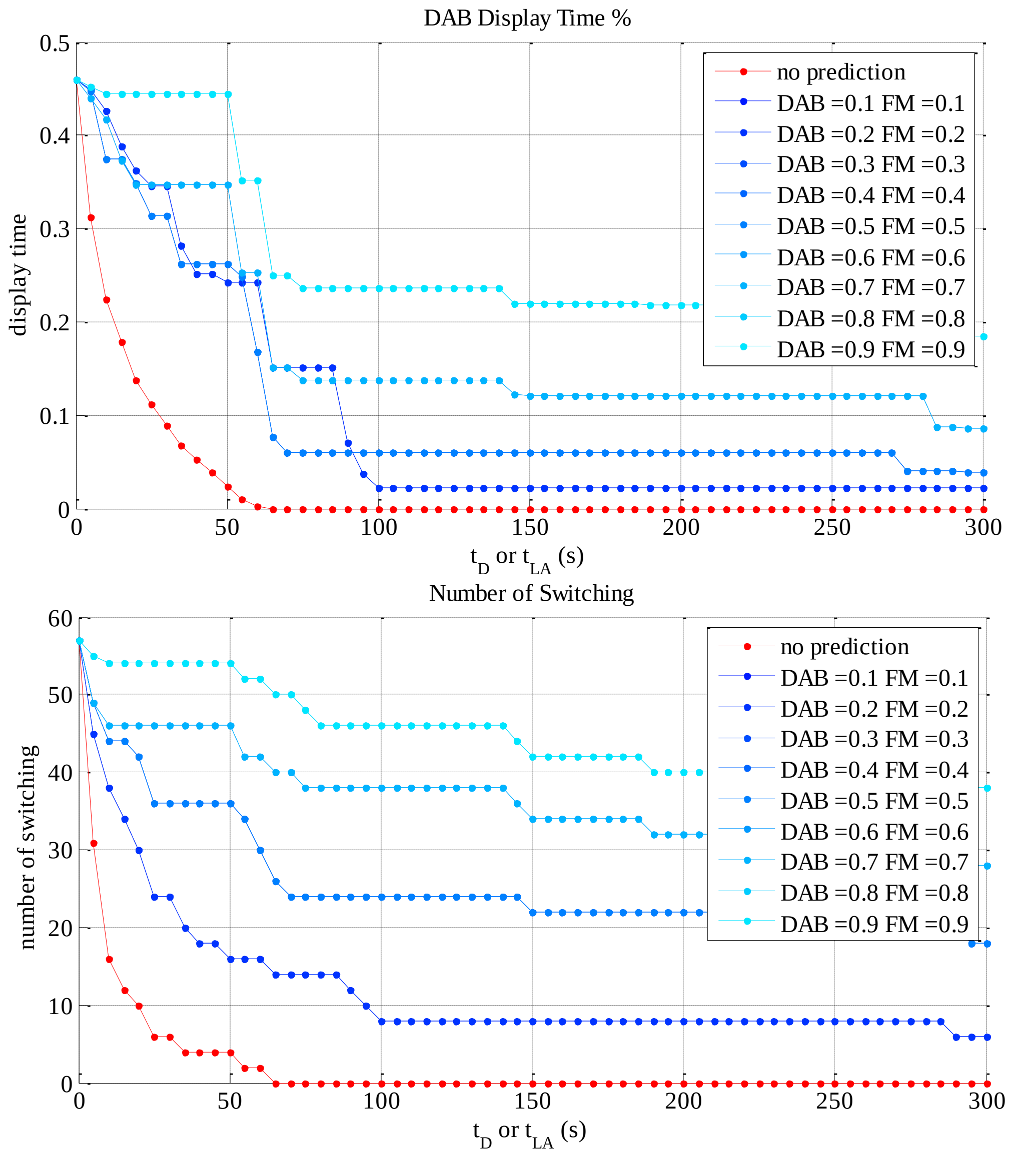



Figure 5.8: Performance of algorithm with 2013-03-06 Memingen to Biberach

[12] The value of threshold in the figure means the determinant value for quantization, e.g. when FM or DAB threshold = 0.3, than the probability which is larger than 0.3 should be considered as receivable and the one less than is not receivable.

For the example above, the threshold 0.5 might be the best since it maximize the DAB display time with relative excellent reduction of switching. However, the threshold changes according to the test drives e.g. the optimal threshold for the example in figure 5.9 below should be 0.3.

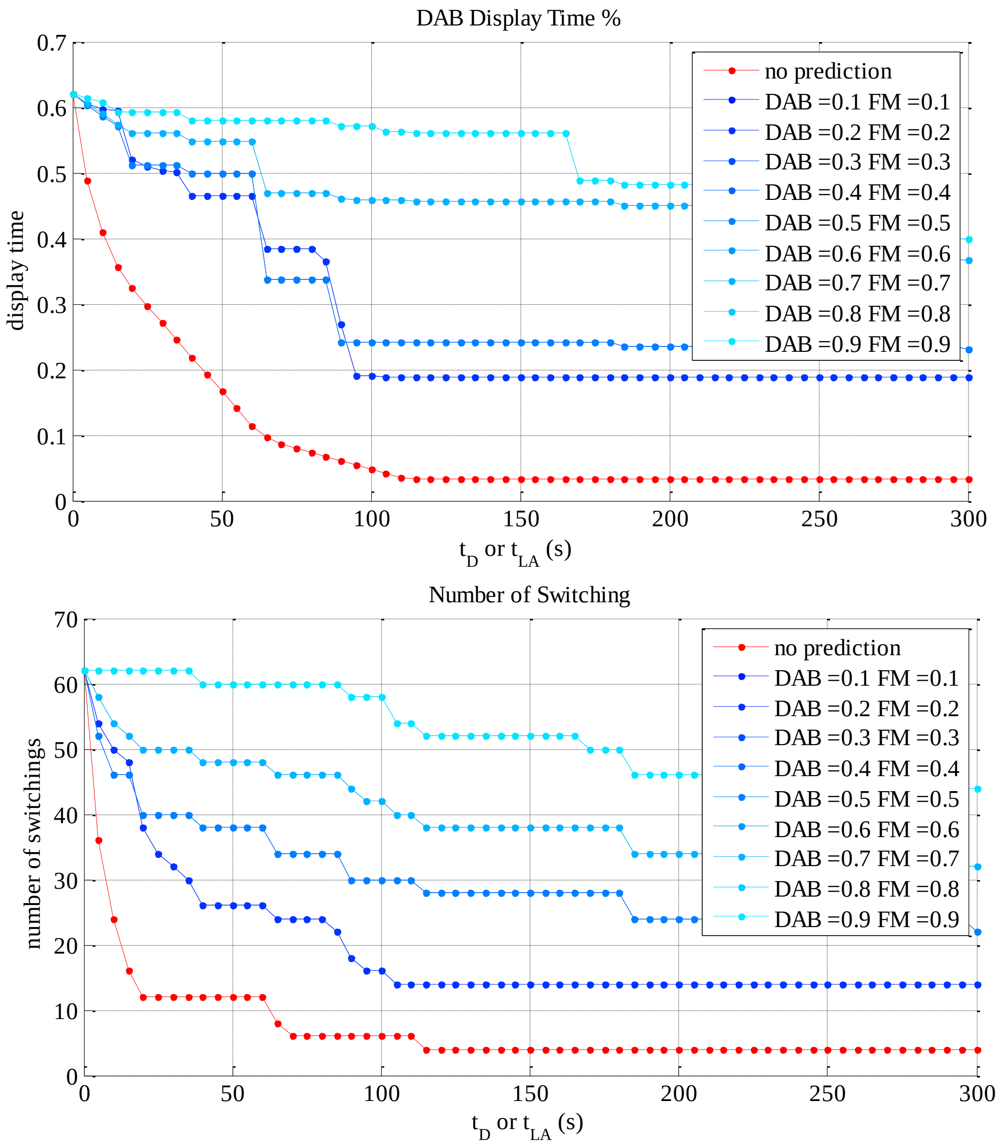



Figure 5.9: Performance of algorithm with 2013-03-15 Memingen to Biberach

The contradictive result signifies that the new algorithm with fixed prediction threshold is dissatisfactory. As the prediction differs with an unsuitable threshold, the performance becomes worse, because a wrong prediction only makes the decision state inaccurate.

## 5.5 Algorithm with adaptive threshold without history

When we consider the correlation between the sample points on the route, it is always the neighbor points maintain the strongest correlation compared with the others. The more distance from one single point to the other, the less influence for the prediction. In this section, we use the simplest case by using the current point's prediction value, as the threshold applied for the next given time period.

It is known from last section that an improper prediction threshold causes even worse performance. Since a fixed threshold cannot be set, an adaptive threshold, which adjusts itself, is needed during the simulation.

Every time when the tuner meets the decision making whether to switch from FM to DAB, it first takes the probability of DAB receivable at current location, e.g. 35%, then looks ahead for a certain time T in the prediction data by applying this probability value. If in T the DAB receivable probability remains always higher than 35%, the tuner considers there is stable DAB reception in the next T and switches to DAB. Otherwise the tuner should stay on FM.

Compared to the algorithm with fixed threshold in section 5.4, the difference is the procedure to determine DAB receivable or not according to prediction as figure 5.10 explained. By the algorithm upside, the threshold never changes after defined; the one downside takes the probability value as the current threshold and applies it to the next given lookahead time period.

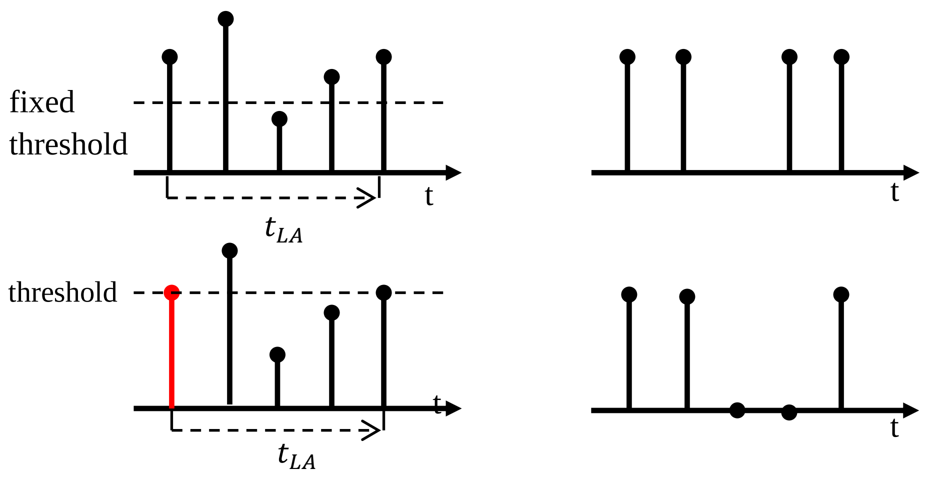



Figure 5.10: Determination using fixed threshold (up) and current probability as threshold (down)

### 5.5.1 Algorithm model

The only change in the algorithm is the “Thresholding” inside the DAB enable logic (figure 5.7). In the algorithm in section 5.4, the threshold value of prediction is pre-defined. In the new algorithm model, the logic -“Thresholding” needs no pre-defined threshold. So compared to the algorithm with fixed threshold, there is no input of “fixed threshold”.

### 5.5.2 Simulation result

As the previous discussion, we use the selected 12 test drives compared with original algorithm to examine algorithm performance. The look-ahead time 30s is as example value being chosen.

Table 5.5: Performance of adaptive threshold algorithm without history with look-ahead time 30s

| Test Drive | DAB Time | Total Switch | Switch < 5s | Switch < 30s | Switch < 180s | Switch < 600s |
|---|---|---|---|---|---|---|
| 2012-08-15 S to U | 92.05% | 25.98 | 7.94 | 16.60 | 22,36 | 23.81 |
| 2013-01-21 S to U | 40.72% | 37.17 | 9.29 | 17.03 | 30.97 | 35.62 |
| 2013-02-19 S to U | 94.46% | 15.36 | 3.54 | 8.27 | 12.40 | 14.77 |
| 2012-08-31 U to S | 76.97% | 31.49 | 4.97 | 15.74 | 27.34 | 29.83 |
| 2013-02-07 U to S | 53.88% | 54.81 | 17.20 | 32.24 | 50.51 | 52.66 |
| 2013-02-15 U to S | 71.33% | 30.92 | 1.99 | 11.97 | 28.93 | 28.93 |
| 2013-03-06 B to M 1 | 79.86% | 132.00 | 76.01 | 108.01 | 128.02 | 128.02 |
| 2013-03-06 B to M 2 | 85.57% | 141.74 | 85.05 | 125.54 | 137.69 | 137.69 |
| 2013-03-15 B to M 1 | 65.12% | 116.87 | 42.50 | 74.37 | 111.55 | 111.55 |
| 2013-03-06 M to B 1 | 23.08% | 96.18 | 26.71 | 64.11 | 90.82 | 90.82 |
| 2013-03-06 M to B 2 | 67.90% | 137.42 | 42.65 | 104.25 | 127.94 | 132.68 |
| 2013-03-15 M to B 2 | 47.26% | 141.00 | 61.16 | 89.38 | 136.43 | 136.43 |

From table 5.5 above compared with the original performance, the fast switchings are also greatly reduced. For example the route between Memingen and Biberach, the number of fast switching (<5s) is around 180 per hour in the original algorithm. In our case, the number of fast switching sinks to average 50 per hour. On the other hand, compared to the algorithm without prediction in section 5.3 (see table 5.3), this new algorithm can preserve more DAB time.

The performance comparison regarding of DAB display time and number of switching is shown in figure 5.11 below. From the picture it is obvious that new algorithm can preserve more DAB display time than the one without prediction. When the chosen look-ahead time is small, the number of switching from the new algorithm is much worse than the one without prediction, see when delay around 20s in figure 5.11, which indicates probably an unsatisfactory reduction of fast switching.

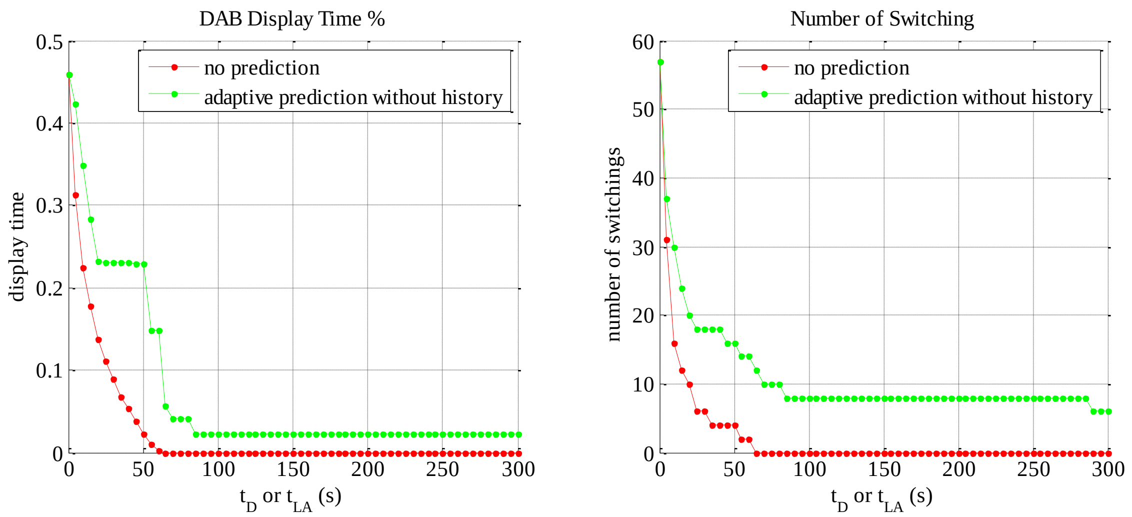



Figure 5.11: Performance of algorithm in route 2013-03-06 Memingen to Biberach[13]

Compared the new algorithm with the algorithm without prediction, we see that by scarifying the same DAB display time, the new algorithm can reduce more fast switching than the one without prediction as figure 5.12.

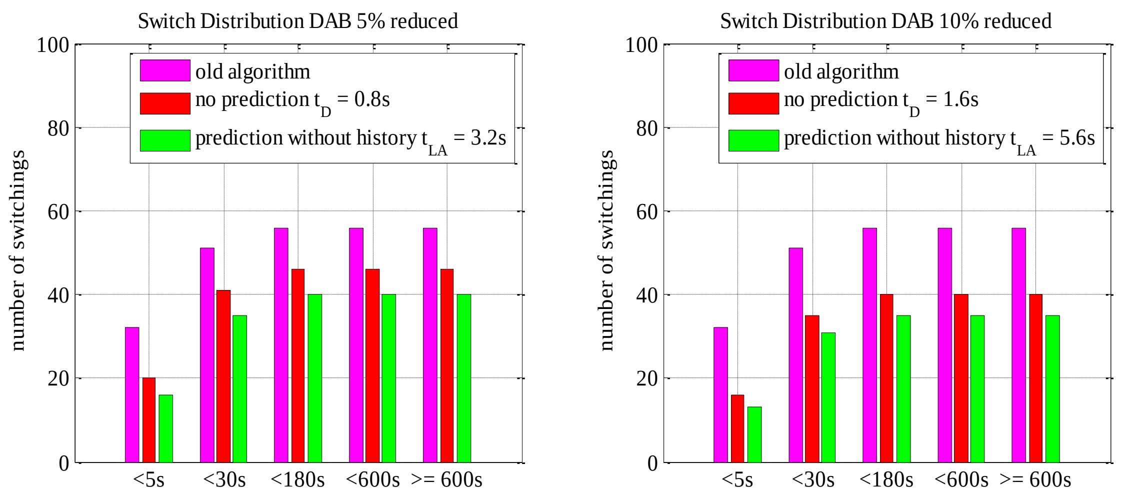



Figure 5.12: Fast switching in route 2013-03-06 Memingen to Biberach 2 comparison with the same DAB display time sacrifice[14]

[13] The time in x- axes represents differently in algorithm in section 5.3 and algorithm here. The time in algorithm section 5.3 – without prediction, means delay; the one in this section represents time looking ahead to the future, which has the same meaning in the next coming algorithms with prediction technique.

One more example for the performance comparison is as in figure 5.13 followed. Since the route between Memingen and Biberach is in area with continuous weak signal reception, the one between Stuttgart and Ulm in which the reception stays relative stable is also of our interest. Figure 5.13 below illustrate an excellent operation within route Ulm to Stuttgart. They are similar as performance in figure 5.11 and 5.12. Please notice that in some cases this algorithm can be worse (example in figure 5.14 on the left side) because the number of fast switching is more than the algorithm without prediction.

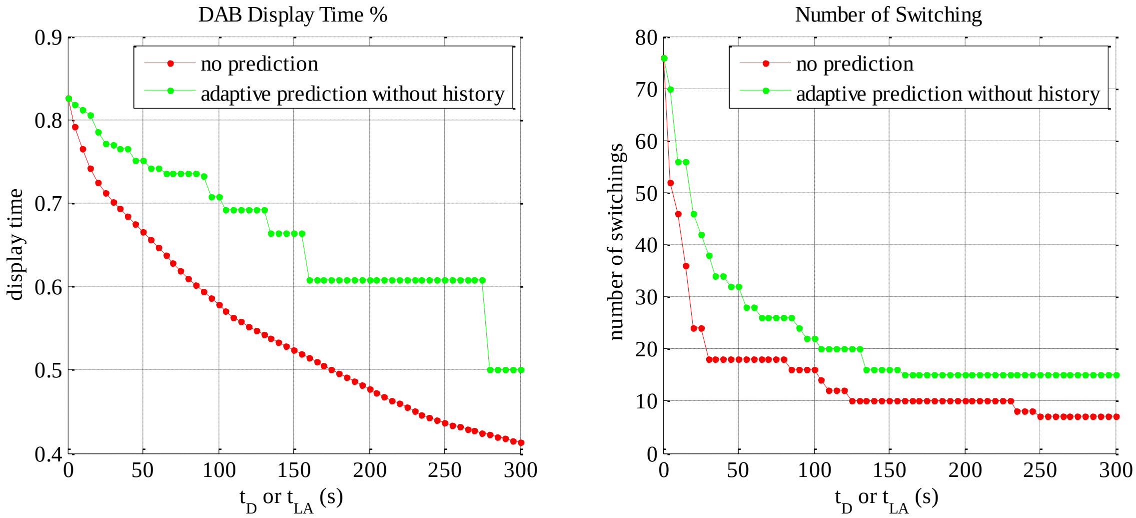



Figure 5.13: Performance of algorithm in route 2012-08-31 Ulm to Stuttgart

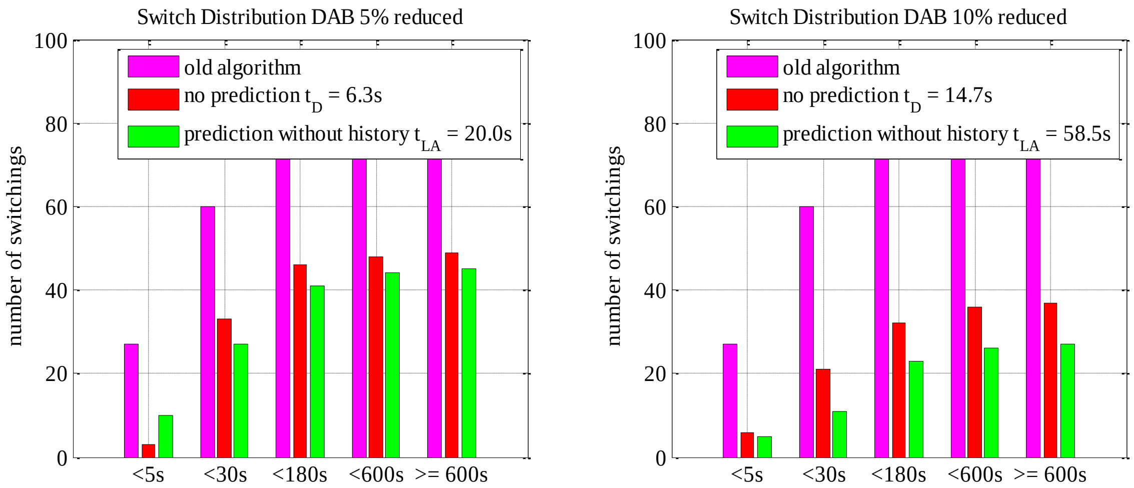



Figure 5.14: Fast switching in route 2012-08-31 Ulm to Stuttgart comparison with the same DAB display time sacrifice

[14] The number at each algorithm, e.g. no prediction at 0.8 on the left side of the picture, means at 0.8s delays the DAB display time by algorithm without prediction has 5% reduction.

## 5.6 Algorithm with adaptive threshold with history knowledge

From section 5.5, we discussed the availability of using an adaptive threshold for prediction. The analysis result is promising, which leads to another assumption whether with more history knowledge provided would make the prediction even better.

As the prediction data is considered as a series of inputs in time axes, to determine a certain value in the current point by the historical inputs and current input is mathematically a discrete casual series in filter theory. There are plenty of filters to find optimal value of the current input with historical inputs. However, it is not the key point for us to compare performance of each different prediction under several filter methods. It is therefore only to use a simplest way with fixed length of the historic data – like a window function, to combine them with the current input.

Unlike the idea in the algorithm in section 5.5 (see figure 5.10), the threshold decision considers the value of current prediction, and also with prediction in the past[15]. The idea of taking historical measurement is that if there is a threshold, which makes the prediction data most like the known historical measurement, then this threshold might be also applied to the future and makes the quantized prediction fit to the real scenario. The way to judge the similarity of prediction and real measurement is by applying the mean square error as the equation 5.1 below. Here x is the prediction data, y is the real measurement. The less the result is, the more similar the data is.

$$Error = \frac{1}{n}\sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2 + \cdots + (x_n - y_n)^2} \tag{5.1}$$

For example in figure 5.15, the system needs a threshold for the determination within the look-ahead time. Since it has the measurement data during the drive, it looks back to the data and compares them with the prediction. In the past time, the measured data is 10000, which means at point A the car received DAB service and from B to E the car did not. If with the threshold 1, then the prediction after thresholding is 11011(as order ABCDE), which makes the error is 1.73. Furthermore, if with the threshold 2, then the prediction after thresholding is 10000, which makes the error is 0. Therefore the system chooses the threshold 2 to determine the prediction data. In the example, the threshold 2 is applied to the period within the look-ahead time, and according to the threshold 2, the prediction data in this period is 10101, which contains 2 times DAB service loss. Therefore in this situation, the system waits in FM state. Actually the assumption if the prediction threshold fits to the previous measurement then it fits to the future as well is not always true. In this algorithm, we consider that the situation in the future is similar as our previous experience[16].

[15] We call it prediction in the past because the car has driven through these geo-location points already.

[16] The previous experience means the previous test drives we took for the same route.

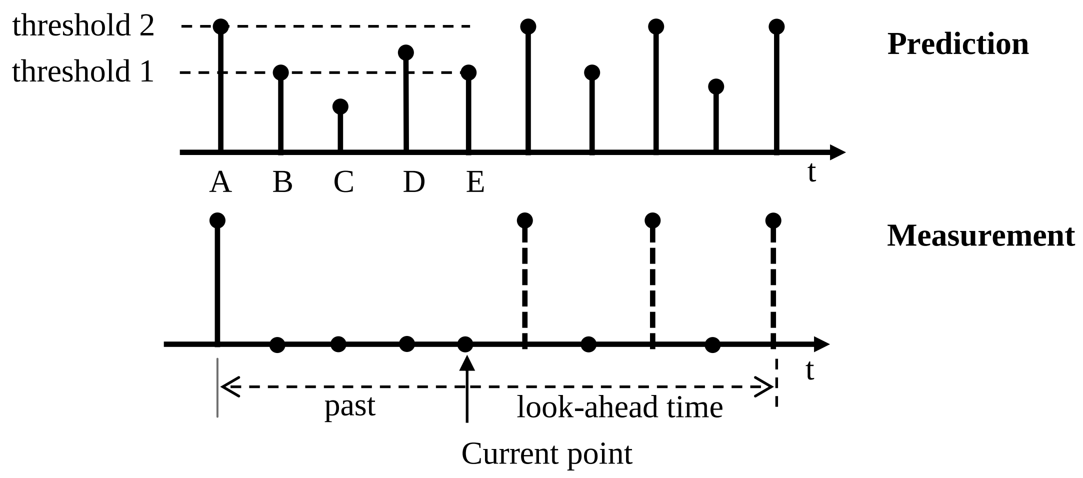



Figure 5.15: Threshold decision with prediction (up) and measurement (down)[17]

In the simulation model, we applied 30s as the time the system looks into the past. Furthermore, in order to simplify the simulation procedure, the threshold will be tested from 0.1 to 0.9[18] with step 0.1, in order to find the one which makes the result in equation 5.1 minimal.

### 5.6.1 Algorithm model

The DAB enable logic in the simulation model also changed in this case. Since the historical knowledge also is used, an extra input should be added to the timer logic as figure 5.16 below.

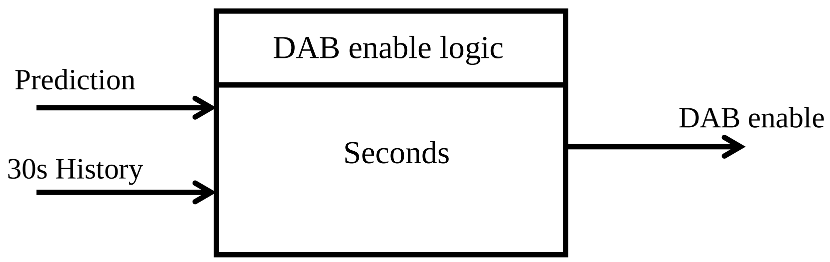



Figure 5.16: DAB enable logic with history knowledge input

[17] The points after current points are with dotted line presented because they are unknown data in the future.

[18] We use 0.1 as step, because in the simulation maximal 10 test drives are applied for each route.

The internal structure of the enable logic is shown as in figure 5.17, in which the thresholding logic contains necessary history information as one of the inputs.

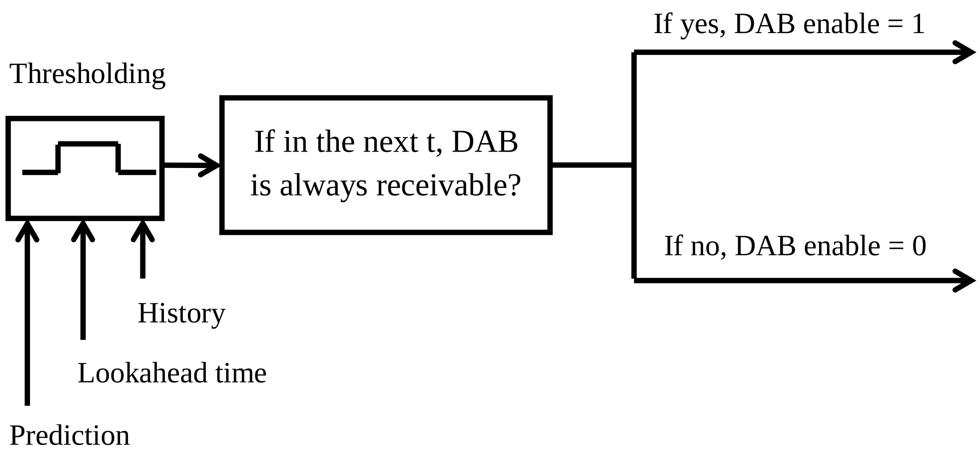



Figure 5.17: Internal logic of new DAB enable logic

### 5.6.2 Simulation result

Table 5.6 with selected 12 test drives shows the new algorithm's performance compared with original algorithm. Here the look-ahead time 30s is taken as example.

Since the algorithm considers historical data, it should maintain more accurate threshold calculation[19] than the one in section 5.5. With a more precise prediction, the number of switching should be more reduced. With the same delay, the algorithm with history can generally reduce more number of switching compared with the algorithm without history knowledge. See figure 5.18 below.

When we compare the different algorithms, the one with history knowledge can reduce more number of switching with DAB time 5% and 10% loss. But sometimes it is worse in decreasing the fast switching, see figure 5.19. In this example, the new algorithm has worse performance in the fast switching reduction than the algorithm in section 5.5.

[19] The prediction data for each algorithm is the same. The different thresholding methods make performances not identical.

Table 5.6: Performance of adaptive threshold algorithm with history with delay 30s

| Test Drive | DAB Time | Total Switch | Switch < 5s | Switch < 30s | Switch < 180s | Switch < 600s |
|---|---|---|---|---|---|---|
| 2012-08-15 S to U | 93.33% | 27.42 | 7.94 | 19.48 | 23.81 | 25.26 |
| 2013-01-21 S to U | 47.28% | 49.55 | 9.29 | 30.97 | 43.36 | 48.01 |
| 2013-02-19 S to U | 95.42% | 20.09 | 4.73 | 12.40 | 17.13 | 19.49 |
| 2012-08-31 U to S | 81.85% | 59.66 | 20.71 | 46.40 | 56.34 | 58.00 |
| 2013-02-07 U to S | 60.78% | 78.45 | 26.87 | 58.03 | 75.23 | 76.30 |
| 2013-02-15 U to S | 71.33% | 34.91 | 3.99 | 16.96 | 32.92 | 32.92 |
| 2013-03-06 B to M 1 | 74.18% | 52.01 | 16.00 | 28.00 | 48.01 | 48.01 |
| 2013-03-06 B to M 2 | 79.64% | 44.57 | 4.05 | 20.25 | 40.50 | 40.50 |
| 2013-03-15 B to M 1 | 66.59% | 127.49 | 42.50 | 90.31 | 122.18 | 122.18 |
| 2013-03-06 M to B 1 | 29.68% | 74.80 | 16.03 | 37.40 | 69.40 | 69.40 |
| 2013-03-06 M to B 2 | 56.32% | 61.06 | 0.00 | 28.43 | 52.12 | 56.86 |
| 2013-03-15 M to B 2 | 44.69% | 84.68 | 18.82 | 37.64 | 79.97 | 79.97 |

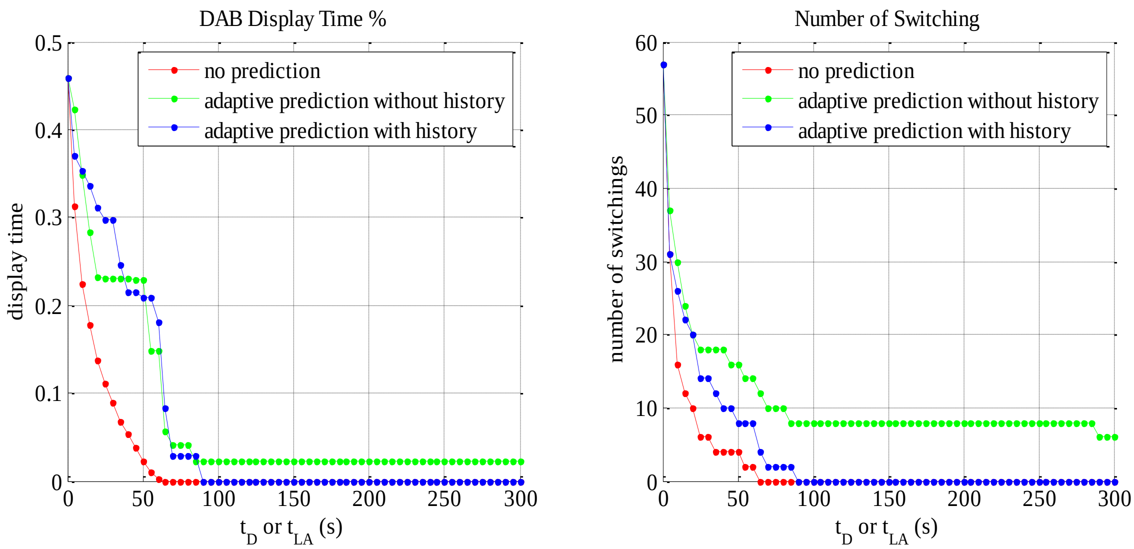



Figure 5.18: Performance of algorithm in route 2013-03-06 Memingen to Biberach

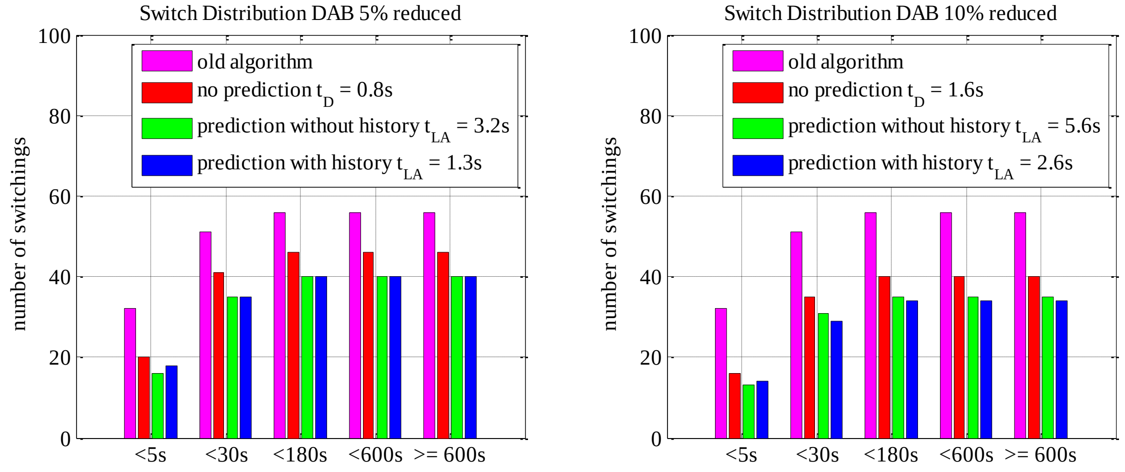



Figure 5.19: Fast switching comparison in route 2013-03-06 Memingen to Biberach with the same DAB display time sacrifice[20]

Another example from Ulm to Stuttgart is shown in figure 5.20 and figure 5.21 as route with stable reception. We see the new algorithm's performance in this example is much worse than the one without history. Therefore, the new algorithm in this specific case should not be taken.

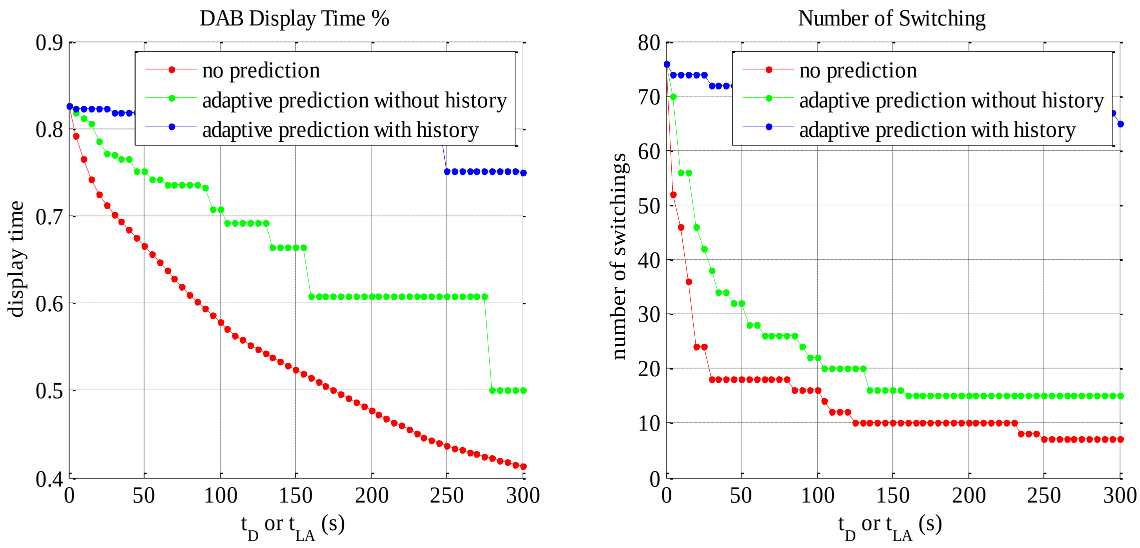



Figure 5.20: Performance of algorithm in route 2012-08-31 Ulm to Stuttgart

[20] The number at each algorithm, e.g. no prediction at 0.8 on the left side of the picture, means at 0.8s delays the DAB display time by algorithm without prediction has 5% reduction.

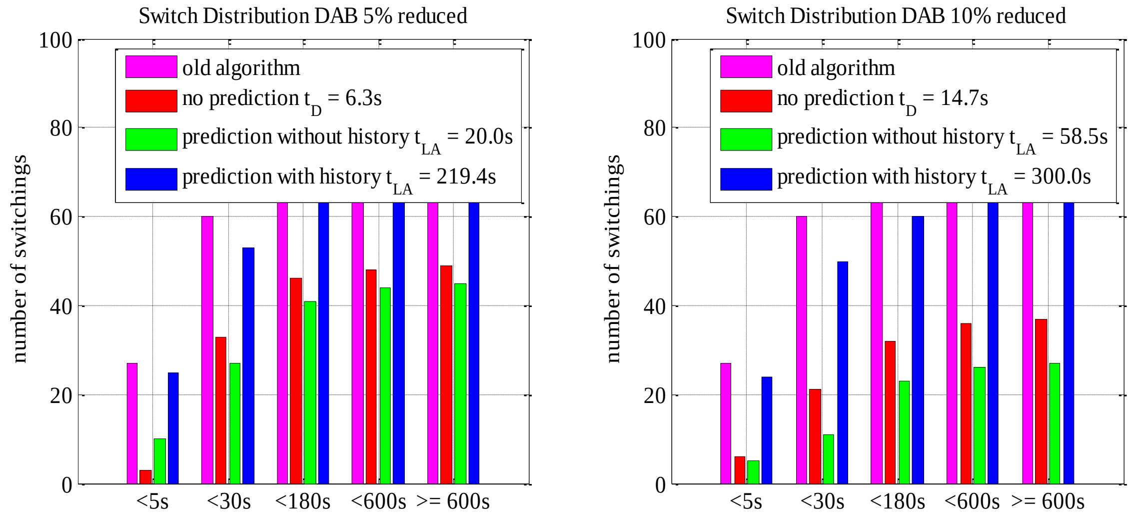



Figure 5.21: Fast switching comparison in route 2012-08-31 Ulm to Stuttgart with the same DAB display time sacrifice

## 5.7 Performance comparison

The previous attempts are shown under different delay/prediction range, in order to find an average time, which fulfills our requirement, we recommend some typical values in table 5.7 respectively under DAB display time sacrifice with 5%, 10% and 20%. These typical values are taken according to the route with dramatic DAB reception vary e.g. Biberach and Memingen. They typically represent the fast switching scenario as the same as part of fast switching in route Stuttgart and Ulm.

Table 5.7: Average delay/prediction length

| **Condition** / **Algorithms** | DAB Reduction 5% | DAB Reduction 10% | DAB Reduction 20% |
|---|---|---|---|
| no prediction[21] | 1.65s | 3.33s | 9.46s |
| prediction without history[22] | 11.60s | 20.9s | 39.27s |
| prediction with history | 2.97s | 10.57s | 53.00s |

The performance parameters like DAB display time, number of switching etc., which are explained in section 5.1, are taken into the list. See the table 5.8, table 5.9 and trable 5.10.

[21] The time is delay time in order to avoid fast switching

[22] The time is length of prediction windows, how much prediction ahead should be taken. It is the same as the prediction with history.

Table 5.8: Performance comparison under average time with DAB 5% reduction

| **Route between Memingen and Biberach** | | | | |
|---|---|---|---|---|
| Algorithms | display % | switching | < 5s[23] | < 30s[24] |
| No prediction | 95.00% | 64.71% | 40.74% | 58.70% |
| Prediction without history | 92.19% | 41.18% | 22.22% | 30.43% |
| Prediction with history | 94.61% | 64.71% | 55.56% | 58.70% |
| **Route between Stuttgart and Ulm** | | | | |
| Algorithms | display % | switching | < 5s | < 30s |
| No prediction | 98.85% | 92.36% | 72.90% | 89.31% |
| Prediction without history | 97.92% | 85.81% | 67.07% | 78.75% |
| Prediction with history | 99.29% | 94.85% | 90.25% | 91.58% |

Table 5.9: Performance comparison under average time with DAB 10% reduction

| **Route between Memingen and Biberach** | | | | |
|---|---|---|---|---|
| Algorithms | display % | switching | < 5s | < 30s |
| No prediction | 87.87% | 51.65% | 21.70% | 43.16% |
| Prediction without history | 81.99% | 50.99% | 38.19% | 42.40% |
| Prediction with history | 86.81% | 45.54% | 29.61% | 35.47% |
| **Route between Stuttgart and Ulm** | | | | |
| Algorithms | display % | switching | < 5s | < 30s |
| No prediction | 96.80% | 81.89% | 38.29% | 73.64% |
| Prediction without history | 95.54% | 76.94% | 56.72% | 64.17% |
| Prediction with history | 98.80% | 91.68% | 83.74% | 86.72% |

Table 5.10: Performance comparison under average time with DAB 20% reduction

| **Route between Memingen and Biberach** | | | | |
|---|---|---|---|---|
| Algorithms | display % | switching | < 5s | < 30s |
| No prediction | 87.87% | 51.65% | 21.70% | 43.16% |
| Prediction without history | 81.99% | 50.99% | 38.19% | 42.40% |
| Prediction with history | 86.81% | 45.54% | 29.61% | 35.47% |
| **Route between Stuttgart and Ulm** | | | | |
| Algorithms | display % | switching | < 5s | < 30s |
| No prediction | 96.80% | 81.89% | 38.29% | 73.64% |
| Prediction without history | 95.54% | 76.94% | 56.72% | 64.17% |
| Prediction with history | 98.80% | 91.68% | 83.74% | 86.72% |

[23] <5 = Current number of switching less than 5s / original number of switching less than 5s.

[24] <30 = Current number of switching less than 30s / original number of switching less than 30s.

Generally speaking, both algorithms with prediction technique are better in DAB time preservation, which means they can to some extent avoid unnecessary DAB lose compared with algorithm without prediction. On the other hand, the algorithm without prediction can reduce more DAB fast switchings. With the growth of collected data, which is the experience for certain route, the prediction can better describe the reception in the route.

With the given assumption we grant an overview of performance when prediction information is perfect in figure 5.22 and figure 5.23.

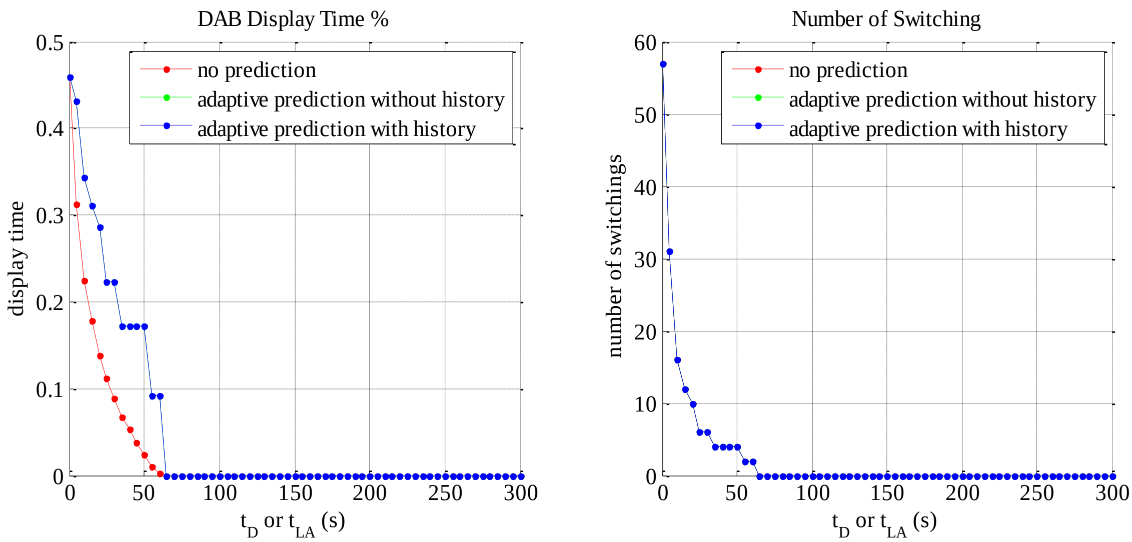



Figure 5.22: Performance with perfect prediction in route 2013-03-06 Memingen to Biberach

From figure 5.21, two points are of our attention: 1) DAB display time with growth of time looking into future ahead decreases the same in both algorithms with prediction technique. 2) Three curves in number of switching reduction are the same. For point 1, because the prediction data is 100% the same as the test drive data, which means with or without history knowledge plays no role to make the prediction more compatible to the reality after thresholding. Therefore, two of them get the same DAB display time. For point 2, three of them decrease the fast switchings, which means the performances of fast switching reduction for them are the same, since the algorithms with perfect prediction knows exactly when fast switchings occur. And for the slow switchings, three of them will not reduce them.

Hence, the algorithms with prediction could reduce as many as the one without prediction, when more knowledge gained from the route.

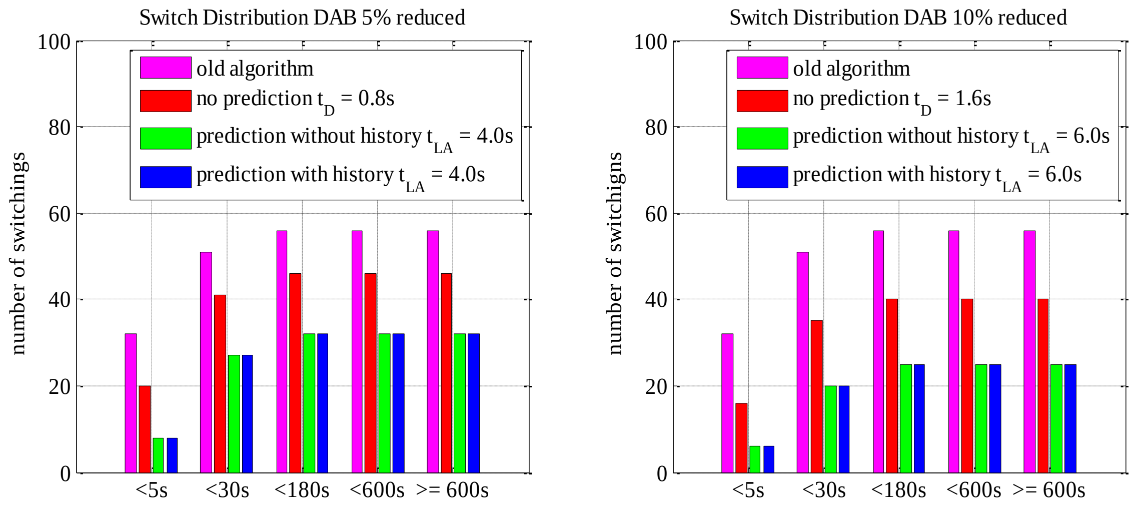



Figure 5.23: Fast switching comparison with perfect prediction in route 2013-03-06 Memingen to Biberach with the same DAB display time sacrifice

Figure 5.22 also shows the much better performance when the algorithms have perfect knowledge of the route. From the discussion of figure 5.23, we see the performances for both algorithms with prediction technique are the same. One more example for route from Ulm to Stuttgart is also shown as figure 5.24 and figure 5.25 below.

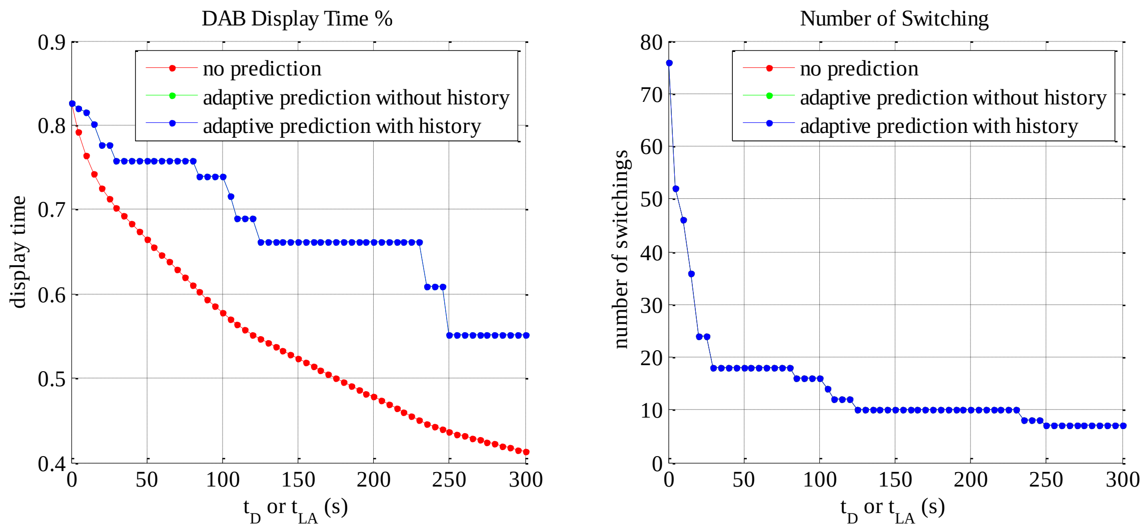



Figure 5.24: Performance with perfect prediction in route 2012-08-31 Ulm to Stuttgart

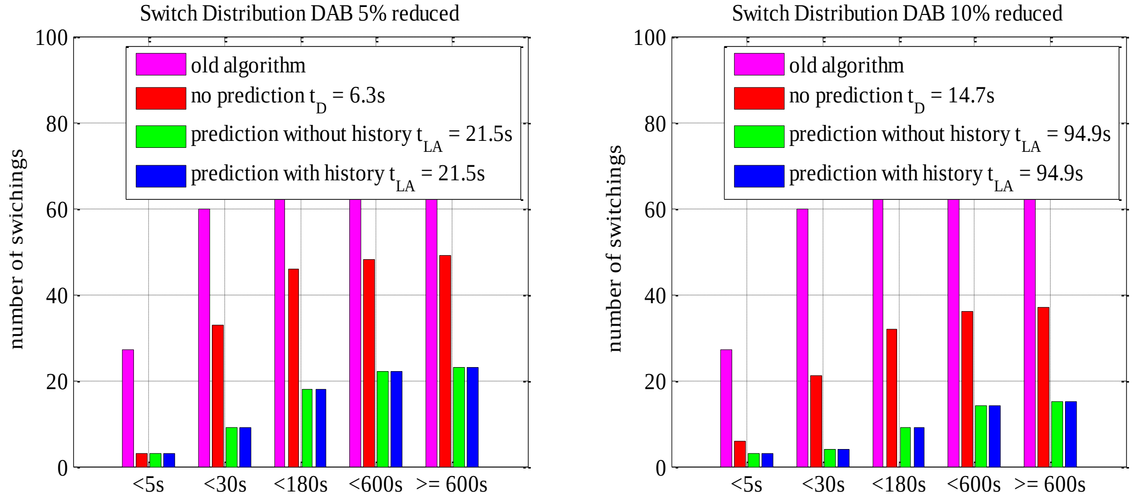



Figure 5.25: Fast switching comparison with perfect prediction in route 2012-08-31 Ulm to Stuttgart with the same DAB display time sacrifice

## 5.8 Tile-based algorithm in practical application

In the practical application, the prediction data is not continuously provided due to some reasons. One is the economic purpose; one is the huge amount of update from each car. It means with the promising performance of the algorithm with prediction, we can hardly implement them currently in our car system since the prediction value is set into tiles with approximately 1000m * 1000m large. This will decrease prediction accuracy additionally compared with continuous prediction discussed before.

To overcome the dilemma, another algorithm which utilizes the experience knowledge of the reception on the route in each tile is taken into consideration. The basic idea behind is that, in one tile, there is only one value represented as prediction. See figure 5.26 below. In the previous discussion in this chapter, our prediction technology is made with much more prediction points, which are almost continuous along the route. In the example below, from A to B there are only approximately 11 values.

We also notice that mostly in the real scenarios, the fast switching happens in just specific area compared to the whole drive. And fast switching reduction is our main task in this thesis. Therefore, we can introduce an adaptive delay according to the prediction in each tile.


Figure 5.26: User (as the white point) at area C has only one prediction value

### 5.8.1 Algorithm model

Since we have introduced a changeable delay rather than fixed one, the DAB enable logic should be able to decide, which delay to use in each tile according to the prediction. Different from the previous algorithm model with prediction, the task of prediction only affects the length of delay.

Hence, the timer logic contains still two inputs as in section 5.4. See figure 5.27.

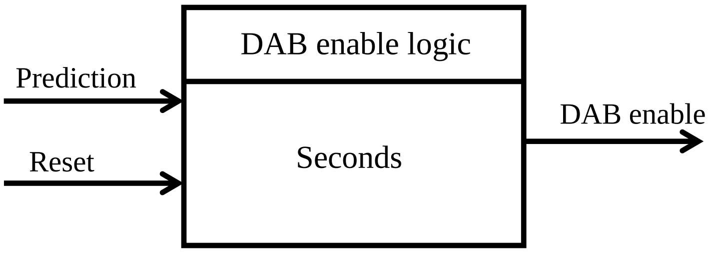



Figure 5.27: DAB enable logic's inputs and outputs of the tile-based algorithm

The enable logic should contain the knowledge of which tile has many fast switching, in order to assign a reasonable length of delay according to that. See figure 5.28.

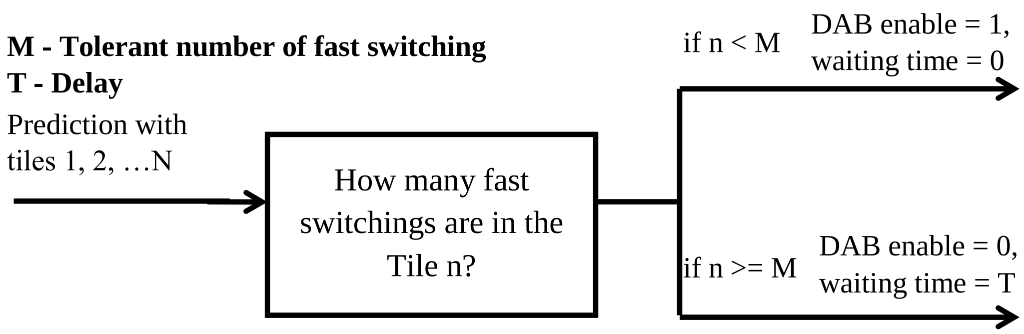



Figure 5.28: Internal logic of DAB enable logic of the tile-based algorithm

In our simulation, there are two possible delays, e.g. 0s and 9s. They indicate the fast switching situation in each tile. With relative more fast switchings, the delay is 9s. Then with less the delay is 0s. This 9s delay is derived from the discussion in section 5.7. The delay, which causes average 20% DAB time loss, is taken as example.

The prediction of number of fast switching in each tile is updated by each test drive gradually. The test drive would use the prediction generated by all of the previous test drive data. For example, the first test drive used no prediction; the second test drive used the first test drives data; the nth test drive used the (n-1)th test drive data. This behavior is more likely the real scenario how the application implemented with a prediction calculated inside the car. The user with more attempts on the same route, on his further drive he is able to use all of the previous experience to improve his drive.

We take the number of fast switching, which is less than 5s, as the value in each tile. After combination of test drive, if the number of switching is larger than 0.1 in one tile, it is considered as area, which should be assigned a delay with 9s. The choice of 0.1 is for simulation purpose. Since there are only 10 test drives taken as example, in order to assign more weight to each drive, the threshold is therefore set to 0.1.

### 5.8.2 Simulation result

We take the route from Biberach to Memingen again to see the performance. All performance indicators agreed are taken into account. We compared the performance improvement with the algorithm with only fixed delay discussed in section 5.3.

From the performance in table 5.11, we compare the two algorithms' performance with the traditional algorithm. At the beginning the performance of tile-based algorithm has no improvement since it has no experience in the route. After several drives, it becomes better compared with the one with only fixed delay. The tile-based one is able to preserve more DAB display time with nearly the same fast switching reduction. See one more example as the way back from Memingen to Biberach as table 5.12.

Table 5.11: Performance comparison between tile-based algorithm and algorithm with longer delay in route Biberach to Memingen

| Test drives | | DAB time % old/new | | Total switching old/new | | < 5s old/new | | < 30s old/new | |
|---|---|---|---|---|---|---|---|---|---|
| 2013-03-06 1 | A[25] | 85.23% | 85.23% | 51 | 51 | 33 | 33 | 46 | 46 |
| | B[26] | | 71.93% | | 15 | | 0 | | 6 |
| 2013-03-06 2 | A | 86.94% | 76.67% | 49 | 9 | 35 | 0 | 46 | 2 |
| | B | | 76.23% | | 9 | | 0 | | 2 |
| 2013-03-15 1 | A | 76.03% | 63.60% | 61 | 24 | 38 | 6 | 55 | 17 |
| | B | | 58.31% | | 20 | | 1 | | 13 |
| 2013-03-15 2 | A | 67.27% | 55.32% | 51 | 15 | 27 | 3 | 46 | 8 |
| | B | | 54.73% | | 15 | | 3 | | 8 |
| 2013-03-15 3 | A | 54.51% | 43.38% | 51 | 21 | 29 | 5 | 45 | 13 |
| | B | | 39.76% | | 19 | | 3 | | 11 |

Table 5.12: Performance comparison between tile-based algorithm and algorithm with longer delay in route Memingen to Biberach

| Test drives | | DAB time % old/new | | Total switching old/new | | < 5s old/new | | < 30s old/new | |
|---|---|---|---|---|---|---|---|---|---|
| 2013-03-06 1 | A | 32.94% | 32.94% | 57 | 57 | 32 | 32 | 51 | 51 |
| | B | | 16.97% | | 18 | | 3 | | 11 |
| 2013-03-06 2 | A | 68.28% | 58.13% | 63 | 25 | 42 | 6 | 58 | 18 |
| | B | | 51.17% | | 23 | | 2 | | 17 |
| 2013-03-06 3 | A | 49.82% | 42.61% | 43 | 13 | 28 | 3 | 38 | 6 |
| | B | | 40.56% | | 9 | | 0 | | 0 |
| 2013-03-15 2 | A | 51.34% | 39.62% | 62 | 26 | 37 | 5 | 54 | 17 |
| | B | | 35.10% | | 26 | | 6 | | 17 |
| 2013-03-15 3 | A | 49.37% | 37.90% | 47 | 23 | 23 | 4 | 40 | 14 |
| | B | | 35.57% | | 21 | | 3 | | 12 |

From the performance comparison between Biberach and Memingen, the tile-based algorithm shows its better performance, as it can maintain more DAB time with nearly many fast switching reduction. With the more knowledge of the fast switching distribution in each tile, the performance becomes increasingly better.

[25] A: The test drive with tile-based algorithm applied.
[26] B: The test drive with algorithm with fixed waiting time 9s applied.

The issue by the tile-based algorithm is a different fast switching distribution at each test drives. One example is taken place in the route between test drives Stuttgart and Ulm. Relevant performance comparison is shown in table 5.13. From the table we see the similar operation as the route in table 5.11 and table 5.12. With more experience collected in the route, the tile-based algorithm performs better and better in fast switching reduction compared to the algorithm with fixed waiting time. However, we notice that in the row 2013-02-19, the tile-based algorithm performs worse than what we expected. By taking consideration of the switching distribution for the five test drives shown in figure 5.29, the area around 3500 and 4000 in route 2013-02-19 contains many fast switchings where the previous four routes do not have. Due to the lack of knowledge of the situation change, the tile-based algorithm performance is worse.

To overcome this issue, we can take more test drives to gain more experience. The other solution could be that all of the tiles are assigned 9s before sufficient experience of the route is achieved. After enough knowledge of the distribution of fast switching along the route is given, the system may switch then some tiles 0s waiting time.

Table 5.13: Performance comparison between tile-based algorithm and algorithm with longer hysteresis in route Ulm to Stuttgart

| Test drives | | DAB time % old/new | | Total switching old/new | | < 5s old/new | | < 30s old/new | |
|---|---|---|---|---|---|---|---|---|---|
| 2013-02-08 | A | 65.64% | 65.64% | 39 | 39 | 13 | 13 | 32 | 32 |
| | B | | 60.92% | | 25 | | 2 | | 16 |
| 2013-02-11 | A | 80.94% | 87.21% | 50 | 46 | 10 | 9 | 34 | 29 |
| | B | | 84.56% | | 44 | | 4 | | 27 |
| 2013-02-13 | A | 94.99% | 92.77% | 42 | 38 | 10 | 4 | 30 | 27 |
| | B | | 91.42% | | 34 | | 2 | | 21 |
| 2013-02-15 | A | 83.87% | 79.93% | 44 | 36 | 6 | 5 | 28 | 15 |
| | B | | 79.20% | | 36 | | 4 | | 15 |
| 2013-02-19 | A | 59.15% | 56.66% | 63 | 52 | 17 | 9 | 43 | 30 |
| | B | | 54.53% | | 46 | | 2 | | 21 |

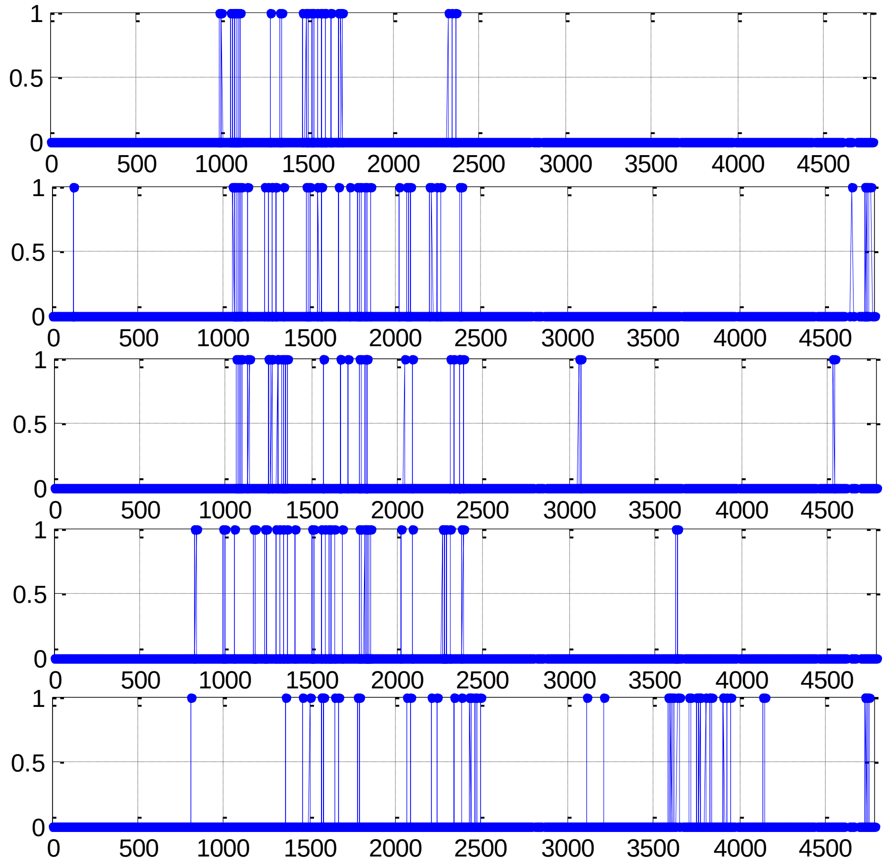



Figure 5.29: Switching distribution in different 5 drives with order from up to down as the same in table 5.13 test drives

## 5.9 Conclusion

In this chapter, we introduced several algorithms with given models. The idea of algorithm without prediction is to prolong the existing delay time so that the number of fast switching is decreased.

A further step to overcome the sacrifice of DAB display time correspondence with the delay in flat switching area, the algorithm with prediction is applied. Two of the algorithms considered prediction inputs are introduced and briefly compared with each performance. One of them considers history data by measurement; the other is not. They are able to provide the more DAB time with much fast switching reduction. Since they require continuous prediction in order to reach good performance, in the practical application, they are not yet of our selection as we cannot provide continuous data in sufficient quality.

## 5.9 Conclusion

Based on the discrete prediction data, which is assigned in each tiles, we let the delay time differ from tiles. In our simulation, we allocate the number of fast switching into every tile to indicate where the area contains lots of fast switching. In the real application, the delay in each tile could be precisely assigned, according to how frequently the fast switching takes place.

# 6 Summary

According to our observation, the traditional algorithm of DAB service following in the car does not reduce the number of annoying DAB to FM switching sufficiently. In some specific situations, e.g. route B312 between Memingen and Biberach, there still occurs fast switchings in average every 20s~30s, which disturbs our customer's listening experience dynamically. In order to provide our user good quality of DAB service and reduce the annoying fast switchings, several algorithms are designed to optimize the traditional performance.

Before simulating the algorithms, the data related to service following were collected. The most important requirement is that the sample data must be accurate. Therefore, the software, which was designed to collect the sample data, is developed by lots of relevant documentation reading and simulation in the laboratory. Besides the theoretical tests, the software is installed in several cars to prove its performance and record the data.

Further in the algorithm model design, we applied a moore state machine for the algorithm simulation. We designed four different algorithms. They are classified as prediction based and non-prediction based algorithms. Particularly for the prediction based algorithm, they are more precisely divided as algorithms with continuous or discontinuous prediction. Each of the algorithms has their advantages and disadvantages.

1. Algorithm with no prediction. It is the simplest algorithm, which introduces a waiting time for switching from FM to DAB. Even though it is not that good at remaining DAB time, it already reduces a lot of fast switchings in most areas. Through plenty of test drives, we recommend an additional delay of 9s, since with 9s more delay, the number of fast switchings in extreme switching scenario is nearly decreased by 80% and reducing DAB time only approximately 20% (see table 5.9 ).

2. Algorithm with fixed threshold prediction. It is the first attempt to utilize prediction to benefit the service following. In this algorithm, the prediction of reception probability over the route is introduced as one of the algorithm input. By setting a certain threshold, e.g. larger than threshold 60% means service receivable, otherwise not receivable. With the prediction data, the car determines when and where to switch. The idea different from algorithm at 1 is that the switching should be made in fast switching areas rather than slow switching areas. After

simulation and test, we found that the optimal threshold depends on the specific route, which means it needs different thresholds in different area. Therefore this algorithm is not worthy due to no common threshold to be applied.

3. Algorithm with adaptive threshold. To overcome the drawback of the algorithm at 2, the adaptive threshold algorithms, which adjust the threshold without pre-definition is needed. The threshold is calculated at each geo-point, according to the current prediction value and measured value. They are designed in two ways, one with history knowledge from the measurement; the other without. They can already reduce lots of fast switching and remain more DAB time than the algorithm without prediction. However, in the real application, these two algorithms are not able to be implemented in our car, since they require continuous and very accurate prediction. Still we simulated the performance of them, when the prediction is perfect. Under this condition, the performance of them is impressively good, which provides the possible development space in the further research.

4. Algorithm with tile based prediction. Since in the real application the prediction data is stored discontinuously, we designed this algorithm with adaptive delay time. Based on the number of fast switchings in each area tile, the delay time is differently chosen. In our simulation, 0s delay time is for tiles where there are no fast switching, and 9s is for fast switching area. This algorithm has a kind of self-learning capability and improves its performance by growth of experience on a certain route. On the other way around, the algorithms could first apply all tiles with 9s waiting time, then dismiss the waiting time later in some tiles, where his knowledge is sufficient. This leads to increasing DAB time with the growth of prediction experience. Its performance is excellent and beneficial. This algorithm shows an improved performance compared to the other algorithms; on the other side, it needs only small overhead. By setting up a small local database, the car is capable to record the fast switching distribution in each tile along the route, which provides the growing knowledge for the prediction. Therefore, based on the current implementation environment, we recommend this algorithm in our car system.

## 6.1 Prospective

With the growing need of customers, the DAB service following currently in the car is facing a challenge to be improved. With the algorithms especially the tile-based algorithm we discussed, the original performance may be positively increased.

In the research work, we use simple and intuitive attempts to analysis the performance. In order to maximize the benefits from prediction data, better technique to get the prediction data should be applied in the further research.

## 6.1 Prospective

The prediction-based algorithm improvement of current automotive technique is not only limited in DAB service following. Furthermore it is possible to apply it in other areas e.g. automotive mobile communication, optimal route planning, etc. In the foreseeable future, this prediction-based technique has still broad and promising space of development.

# Bibliography


[1] P. Ratliff, *Digital Audio Broadcasting Going Firm - The Emerging Standard*, in Terrestrial DAB - Where is it Going?, IEE colloquium, London, 1993.

[2] *ETSI EN 300 401 - Radio Broadcasting System;Digital Audio Broadcasting (DAB) to mobile, portable and fixed receivers*, European Telecommunication Standards Institute, Valbonne, France, 1995.

[3] M. Prosch, *DAB+; The additional audio codec in DAB*, WorldDMB Technical Committe, London, United Kingdom, 2008.

[4] *ETSI TS 103 176 - Digital Audio Broadcasting (DAB);Rules of implimentation; Service information features*, European Telecommunication Standards Institute, Cedex, France, 2012.

[5] *Ins Leere gesendet*, Der Spiegel, Spiegel-Verlag, pp. 94 - 96, 2001.

[6] World DMB, *WorldDMB Global Update*, URL: http://www.worlddab.org/public_document/file/393/WorldDMB-Country_Update-Feb2013-Web__2_.pdf, March 2013.

[7] *ETSI EN 301 700 - Digital Audio Broadcasting (DAB);VHF/FM Broadcasting: cross-referencing to simulcast DAB services by RDS-ODA 147*, European Telecommunication Standards Institute, Valbonne, France, 2000.

[8] Digitalradio, *Empfangneu*, URL: http://digitalradio.de/index.php/de/empfangneu, April 2013.

[9] *MOST Specification Rev 2.4*, MOST Cooperation, Karlsruhe, Germany, 2005.

[10] *NMEA, UBX Protocol Specification*, u-blox AG, Thalwil, Switzerland, 2008.

[11] *Navilock NL-402U USB GPS Data sheet*, URL:

http://www.navilock.de/produkt/60095/pdf.html?sprache=en, April 2013.

[12] *MOST Functinon Catalog*, Daimler AG, Stuttgart, Germany, 2011.

[13] *NMEA 0183 Standard*, National Marine Electronics Association, URL: http://www.nmea.org/content/nmea_standards/nmea_0183_v_410.asp, April 2013.

[14] Y. Gao, *Grabber Documentation*, Daimler AG, Ulm, Germany, 2012.

[15] Y. S. Meng, Y. H. Lee and B. C. Ng, *The Effects of Tropical Weather on Radio-Wave Propagation Over Foliage Channel*, Vehicular Technology, vol. 58, no. 8, pp. 4023-4030, Octobor 2009.

[16] J. E. a. E. Johnson, *Matching, Euler Tours and Chinese Postman*, Mathematical Programming, vol. 5, pp. 88-124, May 1973.

[17] F. H. Zhang, *An algorithm of shortest path based on Dijkstra for huge data*, in Fuzzy Systems and Knowledge Discovery international conference, Tianjin, China, 2009.

[18] J. Kirk, *Open Traveling Salesman Problem - Genetic Algorithm*, URL: http://www.mathworks.de/matlabcentral/fileexchange/21196-open-traveling-salesman-problem-genetic-algorithm, August 2008.

[19] C. Armbrust, D. Schmidt and K. Berns, *Generating Behaviour Networks from Finite-State Machines*, in 7th German Conference on Robotics, Munich, Germany, 2012.

# Glossary

| Term | Definition |
|---|---|
| Head Unit | A component in the car, which provides multimedia service, e.g. TV, Navigation. |
| NTG4.5 | New Telematic Generation is the generation of infotainment system in Mercedes-Benz. |
| Service Information | It provides supplementary information about services, both audio program and data, like station name, service IDs, etc. |
| Fast Information Channel | Part of the transmission frame, comprising the Fast Information Blocks, which contains the multiplex configuration information together with optimal service information and data service components. |
| Fast Information Block | Data burst of 256 bits. |
| Fast Information Group | Package of data used for one feature in the Fast Information Channel, which is available for non-audio related data services, such as paging. |
| Service Linking | The information feature provides service linking information for use when services carry the same primary service component or when the primary service components are related. |
| Hard Link | Services which carry the same primary service component. |
| Soft Link | Services which contain the related primary service component. |
| Ensemble | A set of services transmitted within the same frequency |
| RDS | Radio data system |
| Notification | It is used to inform devices about changes in a property. All devices that are registered for that particular property in the Notification Matrix will receive a status message with the updated value. |
| Notification Matrix | Devices that require status updates when a property changes can register for that particular property in the Notification Matrix. |
| OPType | When accessing function, certain operations can be applied. The type of operation is specified by the OPType (e.g, Set or Get). |
| PI | Program Index, which is the identification of FM services. |

## Appendix 1: Relevant MOST function block

| Field | Description | Note |
|---|---|---|
| Type | Source of Data from tuner, e.g. background or fore-ground. | |
| Name | Station/program name | 1. Written with UTF16 string.<br>2. DAB foreground tuner transmits service with only id but without name string. |
| Content | On show program | Not yet implemented. |
| SvcID | Service unique ID | 1. Only used in DAB service.<br>2. 4 bytes |
| EnsID | Ensemble ID | 1. Only used in DAB service<br>2. 4 bytes |
| EnsName | Ensemble name | 1. Only used in DAB service<br>2. UTF16 string |
| EnsECC | Extended country code of ensemble | 1. Only used in DAB service<br>2. 2 bytes |
| FrequencyTable | Current tuned frequen-cyband | 1. Only used in DAB service<br>2. UTF16 string |
| Frequency | Station frequency | The expression of each frequency in DAB and FM tuner is not the same. |
| Syn | Reception status of the DAB tuner | 0-no DAB signal<br>1-signal detected |
| Mute | Mute status of DAB tuner | 0-no muted, DAB is receivable;<br>1-in contrary |
| SQ | Signal quality, the value depends on the implementa-tion | |
| Counter | Indicates how often the en-semble could not be re-ceived. | 0-service is receivable<br>1-service is not receivable |
| SenderNameIn-fo | Flag indicates attributes of FM station name. | Only used in FM |
| PI | Program index | 1. Only used in FM<br>2. 2 bytes |
| PTY | No explanation due to the lack of documentation | |
| Selected | No explanation due to the lack of documentation | |
| TMC | Traffic information | 0-unknow<br>1-available 2-inavailable |

## Appendix 2: GPS string documentation

The string from GPS receiver is as example:

GPGGA,122528.000,4825.2641,N,00956.5462,E,0,00,99.9,625.6,M,47.7,M,,0000*6C

GPRMC,122529.000,V,4825.2641,N,00956.5462,E,0.00,0.00,240912,,,N*70"

| GPGGA | |
|---|---|
| Flag | GPGGA |
| Time | 122528.000 - 12:25:28 |
| Latitude | 4825.2641 N - 48°25.2641‘N |
| Longitude | 000956.5462 E - 9°56.5462‘ E |
| Fix quality | 0 |
| Satellites | 00 |
| HDOP | 99.9 |
| Altitude | 625.6M |
| Height | 47.7M |
| others | 0000*6C |

| GPRMC | |
|---|---|
| Flag | GPRMC |
| Time | 122529.000 - 12:25:29 |
| Warning | V |
| Laititude | 4825.2641 N - 48°25.2641‘N |
| Longitude | 000956.5462 E - 9°56.5462‘ E |
| Speed | 0.00 |
| Course | 0.00 |
| Date | 240912 - 24.09.2012 |
| Variation | - |
| Checksum | N*70 |

Declaration

All the work contained within this thesis, except where otherwise acknowledged, was solely the effort of the author. At no stages was any collaboration entered into with any other party.

Stuttgart, May 06, 2013 _______________________

(Yuan Gao)