校内放送自動録音システムの開発
—校内放送をより聞きやすくするために—

池上 聡範[*1]・奥村 友陽[*1]・末永 温和[*1]
指導教員：宇井 友寿[*2]・西澤 淳夫[*2]
Email: t-nishizawaat@e.osakamanabi.jp

*1: 大阪府立千里高等学校総合科学科 3 年
*2: 大阪府立千里高等学校

◎Key Words　Raspberry Pi，校内放送，連絡ツール

1. 研究動機

校内放送の重要な連絡を聞き逃して困った経験から、校内放送を自動録音できれば聞き逃しを防止できると考えた。さらに、その録音した音声を文字化して共有するまでを自動化し、常時稼働させるシステムがあればより便利と考えた。また、既存の放送設備に変更を加えずに、後付けで簡単に設置できる装置とすることを目指した。

2. 目的

①校内放送を聞き逃した人が再度、校内放送を確認できるようにすること
②自動文字起こしによって文字情報での校内放送の確認を容易に行えるようにすること
③高い安定性を持つシステムを開発すること
④高い冗長性を持つシステムを開発すること
ここでの冗長性とは、ハードウェアの故障時にシステムが停止することなく動作を継続できるという意味である。

3. 本校の放送システムについて

本校の放送システムは内線電話で校内放送をかけることができる。また、これとは別に放送室や各教科の職員室、会議室などからでも校内放送も行うことができる。このように校内放送をかけられる場所が非常に多い。そのため放送用マイクに機器を設置し、録音するということは現実的でなく、実現することが難しいと考えた。そこで、各教室に置いてある放送用スピーカーの 1 つに機器を設置し、録音することが現実的だと考えた。

4. 本システムの環境について

Raspberry Pi:	小型で低電力のコンピューター
Python3.7:	開発に用いたプログラム言語
FFmpeg:	音源の圧縮に用いたソフト
Slack:	校内放送の共有場所
Discord:	管理用のチャットツール

5. 校内放送の共有について

本システムは校内放送を共有することで再確認を可能とした。共有についての大まかな流れとしては、校内放送を録音する機器（Raspberry Pi）が校内放送を共有するという流れである。共有するものは、校内放送を録音した音声データと自動で文字起こしされた情報である。共有についての詳細は後述する。

6. 本システムの動作の流れについて

本システムは以下の図 1 のように動作する。

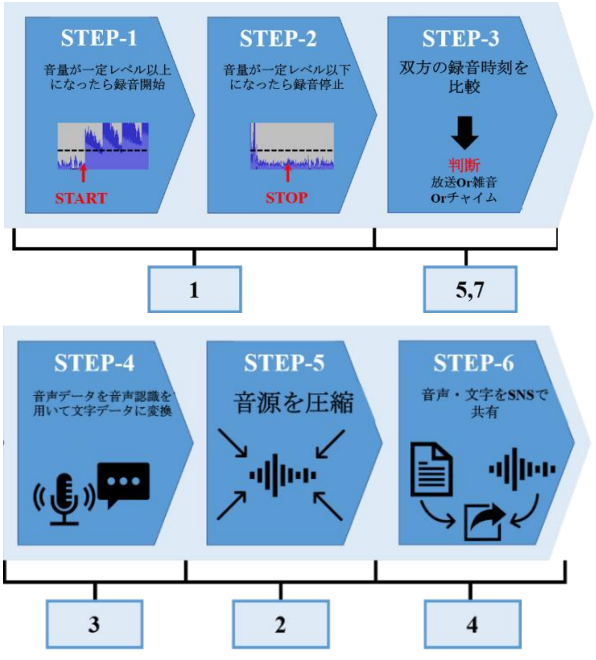

図 1：本システムの動作の一連の流れ
（数字は下記の説明とリンクしている）

7. 搭載されている機能について

7.1. 音声を録音する機能

録音の一連の流れを図 2 にまとめた。

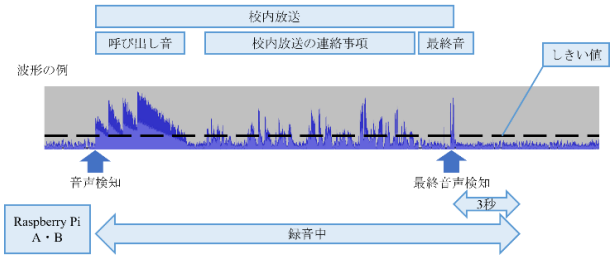

図 2：録音の一連の流れ

この波形は校内放送の一例を表しており、しきい値を超えている初めの波形が校内放送の呼び出すチャイムを表している。このチャイムがしきい値をこえるため、録音が開始される。その後、音量が 3 秒間しきい値を超えない状態が続いた場合に録音を停止する。放送の途中に 3 秒以内の空白があった場合でも録音は継続できる。

7.2. 音声を圧縮する機能

録音された音声は、非圧縮の wave 形式である。元々の音質、音声の長さの関係でそれほどサイズの大きいものではないが、圧縮することで、より素早く Slack への送受信ができるようになると考えた。圧縮には FFmpeg を使用した。FFmpeg とはオープンソースの CLI 上で映像・音声を簡易な編集や変換を行うソフトウェアであり、ライブラリを使用することで Python から操作することができる。以下は実際に使用している引数である（図 3）。

```
ffmpeg -ss 4s -i “入力ファイル”-c:a aac -af = dynaudnorm -ab
22kbps -ac 1 -ar xxxx “出力ファイル”
（実際には ffmpegpython 用の記述である）
```

図 3:FFmpeg の引数

これにより、放送内容の確認が可能な音質で、サイズを圧縮前の 10 分の 1 程度まで圧縮できた。また、録音された音声の先頭 4 秒を切り取ることで、放送開始の呼び出し音をカットし、より素早く放送内容が確認できるようになっている。こうして生成された音声ファイルを http リクエストにより Slack に送信する機構を作成し、音声を検知すると自動で Slack に送信されるようにした。

7.3. 音声認識機能（自動文字起こし）

Google が提供している Google Speech Recognition を利用した。このサービスは多少の誤認識（固有名詞などの漢字表記、似た語の認識ミスなど）はあるものの、校内放送の内容を文字だけで確認できる精度があった。この文字情報も Slack へ送信されるようにした。

7.4. 校内放送の音声・文字情報を共有する機能

共有場所の条件として以下の 3 点を考えた。

・校内放送の音声データを再生できること
・文字情報を表示できること
・生徒と教職員が簡単にアクセスできること

当初は、校内放送の内容をオープンなインターネットに公開せず、校内 LAN のみに共有し、内容を確認できるよう、専用の Web サーバーを立てて IP アドレス指定でアクセスする方法を考えた。また、この方法を採用することで LAN 外からのアクセスを防ぎ、セキュリティーを保つことができると考えた。しかし、令和 3 年度に全生徒に Chromebook が貸与され、それでインターネットを通じて確認できるシステムにすることにした。そこで、私たちは 3 つのチャットサービス LINE, Discord, Slack を比較し検討して、Slack を選択した。その理由は、貸与の Chromebook から接続できたのは Slack のみであったからである。さらに文字データ・音声データの送受信が可能であり、音声データは端末内にダウンロードせずに再生ができる点も適していた。また、ファイルやテキストの送信も可能であった。

7.5. 騒音排除機能

2 台の Raspberry Pi を離れた録音部屋に設置すれば、放送の場合は両方の Raspberry Pi が同時に検知するのに対し、騒音の場合は片方のみが検知するため、2 台の録音の比較により放送と騒音の判別ができるようになると考えた。また、録音開始の音量は非常に大きく、録音機材の近くで大声を出さない限り検知しないので、両方の録音部屋で同時に騒音が検知される可能性は低いと考えた。

2 台の機材をメイン機とサブ機とし、サブ機は検知した音量がしきい値を超えた場合、録音開始時刻をメイン機に送信する。メイン機は自身の録音開始時刻とその情報を比較する。双方の録音開始時刻の差が一定以内であればそれは放送であると判断し、その音声と文字情報を Slack に送信する。

ただし、この処理は 2 台の Raspberry Pi が必要になり、サブ機が故障すると、すべてを騒音として排除してしまう。後述の冗長化を利用することで Raspberry Pi の故障を検知し、故障があった場合は騒音排除機能すなわち録音開始時刻の比較を無効化し、単体で動作するようにする。つまり、故障していない端末が単独でメイン機として動作することになる。

7.6. チャイムによるシステムの故障検知

マイクやスピーカーの故障の場合、双方の Raspberry Pi は生存確認をしているが、録音開始時刻が一致しない（サブ機が音声を検知しない）ため騒音と認識してしまう。そこで私たちはマイクおよびスピーカーの故障検知が必要であると考えた。

チャイムは毎日必ず鳴るため、マイクとスピーカーがともに正常であれば、チャイムを検知するはずである。よってチャイムを検知しなかった場合は、マイクまたはスピーカーが故障していると判断できる。これを利用し、定時のチャイムを検知しなかった場合は、マイクとスピーカーのいずれかが故障していると判断するようにした。

7.7. チャイム排除機能

まず、定時のチャイム時刻をあらかじめ入力しておく。チャイムを検知した時の録音開始時刻と定時のチャイム時刻がほぼ一致すればその音声はチャイムであると判断でき、音声などを共有しない。

しかし、これではチャイムの終了後すぐに放送が入ってしまうと録音が継続されてしまう。この放送がチャイムと誤認識されて排除される可能性がある。そのため、録音した音声の長さを参照し、通常のチャイムの長さより長い場合は放送が含まれていると判断し、Slack に送信するようにした。

また、本校では 2 つの時程（A 時程：45 分授業、B 時程：50 分授業）によってチャイムの鳴る時刻が異なる。ただし、両時程において 8 時 25 分と 8 時 30 分のチャイムは共通しているため、その 2 つのチャイムでマイクの状況を毎日確認する。そして時程により時刻の異なる 1 限終了チャイムによりその日の時程を判断し、その後のチャイム排除は判断した時程によって動作する。

これによりほぼ確実にチャイムを判別し、排除することができた。

7.8. 共有されたデータの自動削除機能

個人情報の保護、容量削減のため音声ファイルと文字情報を一定時間経過すると削除する機能を実装した。本機能は Slack に投稿されたデータを削除するもので、主な動作としては日付が変わった時点で前日に送信されたデータを全消去するものである。

7.9. スイッチによるシステム停止機能

Raspberry Pi 本体に接続されているスイッチを押すことによりその Raspberry Pi の録音処理を停止させることができる。これにより録音教室を授業などで使用する際に、意図しない録音をされることなく使用することができる。

7.10. Discord によるシステムの管理機能

システムの様々な機能を切り替えるために Discord 上からコマンドにより制御をできるようにした。録音の停止やシステムの停止、再起動などが遠隔操作できる。

7.11. システムの冗長化について

システムを安定して動作を継続させる際に、本体の故障の可能性をなくすことはできないと考えた。そこで、故障しても問題の起こりにくい安定的なシステムを開発することにした。そこで、私たちはサーバーなどで用いている冗長化を実装することにした。メイン機（録音および送信処理を行う）とサブ機（7.5 の騒音排除のため）の 2 台の Raspberry Pi を用いて、それらが常に通信している。一方が故障したときはもう一方が管理者にそれを報告する。メイン機が故障した場合はサブ機がメイン機として動作を続ける。この仕組みによって Raspberry Pi の故障が発生しても騒音排除機能を除いて一連の処理を継続できると考えた。この通信はインターネット上で行う。また、LAN ケーブルの断線などにより片方のネットワークが遮断された場合はそもそも生存確認ができないため、故障として判断する。両方のネットワーク故障（ルーターなどによるもの）や停電などではそもそもデータを Slack に送信できないため対策は必要ないと考えた。しかし、停電時の動作は行わなくていいものの、停電による故障の可能性が考えられた。その場合、2 台同時に故障する可能性もあると考えられる。そのため、故障率を軽減させることにした。Raspberry Pi の故障原因の多くを占めるものとして電源の突然喪失による SD カード（Raspberry Pi の全データが入っている場所）の破損があげられる。これは、SD カードの書き込み中に電源の喪失により書き込みがストップし、SD カード内のデータが破損するというものである。そのため、私たちは SD カードを読み取り専用にすることで故障率を軽減させることにした。この際、録音データを SD カードに保存できなくなるため、USB メモリーなどの外部ストレージに保存することにした。それでも一方の SD カードが破損した場合、もう一方が故障を検知して管理者に報告するため、迅速な交換が可能である。

これらの機能により高い安定性を確保できるようになった。

8. 運用実験について

8.1. 第 1 回運用実験

期　　間　2021/12/15-2021/12/18

公開機能　録音・共有・文字起こし

公開範囲　開発者（私たち 3 人）

実験目的　システムが実際に放送を検知し、データの送信まで行うことができるか。

結　　果　問題なく放送の検知・送信ができていた。文字起こしは一部誤認識があったが、精度は予想以上に高かった。また、雑音排除を使用しなかったにもかかわらず雑音の送信はほとんどなかった。

8.2. 第 2 回運用実験

期　　間　2022/01/13-2022/02/03

公開機能　録音・共有・文字起こし

公開範囲　開発者・一部教員

実験目的　第三者に実際に利用してもらいシステムの使用感・改善点などの調査。また、従来の放送システムの使用感・問題点の調査。第 1 回より長期の運用による安定性の検証。

結　　果　おおむね使い心地は好評だったが、チャイム排除機能が未実装で、チャイムのたびにデータが共有される点が改善の要望として挙げられた。

8.3. 第 3 回運用実験

期　　間　2022/04/18-2022/10/05

公開機能　録音・共有・文字起こし・チャイム排除・データ削除

公開範囲　開発者・第 2 回より多くの教員

実験目的　追加した機能の使用感などの調査。より長期間の運用による安定性の検証。

結　　果　追加した機能はおおむね好評だった。長期間の動作により意図しないシステムの停止など安定性に悪影響が発生することがわかった。また、雑音が送信されることもあった。

9. 展望

9.1. 共有手段の追加

現状 Slack のみの共有手段を Google Classroom などでも閲覧できるようにする。前述したとおり Slack はアカウント設定が必要で、生徒および教職員を全員参加させるにはとても手間がかかってしまう。対して Google Classroom は、生徒および教職員の全員が既に参加しており、ユーザー側で新たな設定が不要となる。また将来的には、文字起こしの結果により、共有先の自動振り分けができれば、使用者にとって重要度の高い放送のみの通知を受け取ることなどが可能となると考えている。

9.2. LAN の 2 系統化

現状、本校のネットワーク上で接続されているが、ほかのネットワークを併用して一方で不具合が発生した場合でも一連の処理を継続できるようにしたい。

参考文献

Mac で python の Speech Recognition を使って音声認識ー Qiita
https://qiita.com/seigot/items/62a85f1a561bb820532a
最終閲覧日：2021 年 11 月 6 日
discord.py 入門(1) - Qiita
https://qiita.com/sizumita/items/9d44ae7d1ce007391699
最終閲覧日：2021 年 12 月 11 日
Discord に Webhook でいろいろ投稿する - Qiita
https://qiita.com/Eai/items/1165d08dce9f183eac74
最終閲覧日：2021 年 12 月 11 日