Vol.2023-7 No.3-1
2024/02/27

OpenFlow と AR を用いたコンピュータ通信パケットの可視化に関する提案

浅野海星[†,a)]　宇梶郁[†,b)]

我が国では，近年の教育現場における情報端末の普及や高等学校での情報の授業の必修化に見られるように，情報技術の積極的な活用に留まらず，学ぶ機会についても広がりを見せている．実生活で身近なインターネットに着目すると，インターネット通信の説明には，図や動画などの一方向の表現が用いられてきたが，インタラクティブな表現を用いることで，初学者にとってさらに理解のしやすい表現になるのではないかと考えた．そこで，本稿では，OpenFlow と Augmented Reality を組み合わせることによって，導入のしやすさとインタラクティブな表現を達成するネットワーク通信の新たな可視化手法を提案する．研究室の学生を対象にした検証では，3 ウェイハンドシェイクの様子がわかりやすいといった好意的な意見も得られた．

A proposal for packet visualization of computer communication using OpenFlow and Augmented Reality

Kaisei Asano[†,a)]　and Kaoru Ukaji[†,b)]

In Japan, not only the active use of information technology but also the opportunities to learn are expanding, as seen in the recent spread of information terminals in the educational field and the introduction of mandatory information classes in high schools. Focusing on the Internet, which is familiar to us in real life, one-way expressions such as diagrams and videos have been used to explain Internet communications, but we thought that interactive expressions would make it easier for beginners to understand. In this paper, we propose a new visualization method for network communication that combines OpenFlow and Augmented Reality to achieve ease of introduction and interactive representation. Verification with laboratory students showed positive feedback, such as the 3-way handshake being easy to understand.

1. 序論

1.1 研究背景

我が国は近年，GIGA スクール構想 [1]のもとで，全国の小・中学校に児童生徒 1 人 1 台の PC 端末を急速に整備し，令和 4 年度末時点で，全自治体のうち 99.9%の自治体への整備を達成した[2]．ほとんどの自治体で整備が完了していると言えるだろう．また，高等学校では，令和 4 年 4 月から，情報の授業が必修化されたこともあり[3]，初等・中等教育を通して情報端末や情報技術に触れる機会が数年前と比べて格段に増えた．そこで，この情報端末の普及を活用した，新たな表現を模索することで，学習者の理解を促進することができるのではないかと考えた．

先述した高等学校における情報の科目に着目する．多くの人々が日常で使用するインターネットに関して，高等学校の学習指導要領[3]では，「情報通信ネットワークとデータの活用」の内容において，「情報通信ネットワークの仕組みや構成要素，プロトコルの役割及び情報セキュリティを確保するための方法や技術について理解すること」を求めている．インターネット通信は，高い周波数の電気信号によって通信しているため，通信そのものを見ることは難しい．通信の様子を説明する際に，従来では，図や動画などによる一方向の表現が用いられてきた．しかし，この方法では，実際に通信しているのか直感的にわかりにくい．そのため，生徒が端末を操作して実際に発生した通信をインタラクティブに可視化する表現を用いて通信のやり取りを説明することによって，生徒自身の直感的な理解に繋がるのではないかと考える．

1.2 本研究の目的

本研究では，導入のしやすさと実験環境の再現性を備えた，インタラクティブに体験できるネットワーク通信の新たな可視化手法について提案する．

1.3 先行研究

ネットワーク機器間の通信を可視化する例として，石川らの研究がある[4]．石川らは，複数の Raspberry Pi 同士を繋ぎ，小規模ネットワークを構成した環境上で，各 Raspberry Pi 間に LED テープを設置し，通信が発生した際にノード間に対応する LED を光らせるという手法で通信の流れを可視化した．通信経路の制御には OpenFlow[5]を用いており，L2 スイッチなどのネットワーク専用機器がなくても，身近にある汎用 PC をネットワーク機器として利用できるようにしている．この手法では，今後の課題の項でも述べられている通り，機器の配線が複雑になってしまう問題がある．Raspberry Pi 間を接続する LAN ケーブルに加え，LED テープを配線する必要があるた

† 一関工業高等専門学校 未来創造工学科 情報・ソフトウェア系
National Institute of Technology,ICHINOSEKI College Department of Engineering for Future Innovation,Division of Computer Engineering and Informatics
a) f19001@g.ichinoseki.ac.jp
b) ukaji@ichinoseki.ac.jp

め，配線が煩雑になってしまい，初学者にとってはセットアップが難しい．また，通信させたい機器を新たに増やそうとすると，LED テープを追加する必要があり，機器を増設する際のコストが嵩んでしまうという課題も考えられる．

2. 本論

2.1 手法の提案

通信の可視化をするために，可視化する端末上での「可視化部分」と可視化端末と実験用のネットワーク間を同期する「通信制御部分」の構成で考えた．以下では，それら 2 つを実現する方法について検討する．

可視化部分では，Augmented Reality（以下，AR と表記する）を用いた表現手法を提案する．AR とは，日本語で拡張現実と訳され，カメラで取り込んだ現実世界上に 3D オブジェクトなどのデジタルデータで構成される仮想世界を重ね合わせて表現する技術を指す．AR には，マーカーを使用する方法と使用しない方法の主に 2 つの種類がある．

本研究では，通信する機器の外見に左右されずに，確実に通信機器を検知する必要があるため，紙のマーカーを用いた方式を採用する．使用するマーカーは，特殊なマーカーではなく，一般的な印刷機で作成できるものである．

通信の様子を AR 上で表現することには，以下のようなメリットがある．

- 機器を増やす際には，検知するマーカーを増やすことで対応できるため，必要最低限の配線で構成することが可能
- マーカーは紙に印刷して準備するため，安価で容易に準備することが可能

通信制御部分では，スイッチが通信を受け取った際に，通信が発生したことを可視化端末に知らせる仕組みが要求される．汎用的なスイッチを使用する場合，ポートミラーリング機能を使用して，通信する機器同士のやり取りを可視化端末にも転送する方法が考えられる．しかしこの手法では，可視化端末にパケットキャプチャ機能を実装したり，スイッチごとに転送の設定をしたりと管理の面で煩雑となるデメリットがある．さらに，イーサネットスイッチを準備する必要があり，機器準備の敷居が高くなってしまう．

そこで，パケットの通信制御に OpenFlow を導入することを提案する．OpenFlow の詳細については次項にて説明するが，OpenFlow を導入すると，各スイッチの経路制御を一括で管理することができ，OpenFlow に対応している Open vSwitch[6]を用いれば汎用 PC をイーサネットスイッチとして利用することができるため，前文で挙げた両方のデメリットを解決できる．また，本研究で使用したコントローラーのプログラムを使うことで，他の環境においても，本研究で構築したネットワークと同様のネットワークを容易に構築できるメリットがある．

OpenFlow で定義したネットワーク上を流れる全てのパケットを可視化しようとすると，多くのパケットの様子が表示され，複雑になってしまう．そのため，本研究では可視化する通信に，身近な通信である HTTP 通信を選び，HTTP サーバーとクライアント間のパケットのやり取りを可視化することにした．

2.1.1 OpenFlow

OpenFlow を理解する上で重要な概念である Software-Defined Networking（以下, SDN と表記する）について述べる．

SDN とは，日本語に訳せばソフトウェア定義型ネットワークであり，ネットワークをソフトウェアによって制御する技術全般を指す．SDN は，ネットワークを制御するコントロールプレーンとデータ転送を行うデータプレーンの 2 つで構成される．コントロールプレーンが転送ルールの指示をデータプレーンに与え, データプレーンでは, 与えられたルールに則ってデータを転送する．従来のスイッチは，制御機能と転送機能が 1 つになっていたことにより，機器ごとにそれぞれ経路設定をする必要があった．SDN では，これら 2 つの機能が独立しているため，コントロールプレーンから分散しているデータプレーンを集中的に管理することができる．

SDN を実現するための技術として，OpenFlow がある．OpenFlow では，コントロールプレーンを OpenFlow コントローラーと呼び, データプレーンを OpenFlow スイッチと呼ぶ．OpenFlow プロトコルに対応しているスイッチであれば，OpenFlow スイッチとして動作することができる．コントローラーとスイッチ間は，OpenFlow プロトコルで通信し, コントローラーは, スイッチに対して固有の値であるデータパス ID（以下, DPID と表記）を割り当てる．この仕組みにより，コントローラーがスイッチを一意に識別して集中的に管理している．

OpenFlow スイッチには，通常の L2 スイッチに存在する MAC アドレステーブルが存在しない．その代替として，経路制御するためのフローテーブルと呼ばれるものが存在する．フローテーブルには，特定の条件を満たすパケットに対して行う操作を記述したフローエントリーというものが格納されている．フローエントリーを構成するフィールドであるマッチフィールド部分に，パケットの条件を設定し，インストラクション部分に，そのパケットに対して行う操作を設定する．OpenFlow コントローラーが，このようなフローエントリーを各 OpenFlow スイッチのフローテーブルに遠隔で書き加えたり，削除したりすることによって，集中管理を実現している．全てのフローに合致しなかったパケットは，テーブルミスフローエントリーで指定された処理が行われる．テーブルミスフローエントリーのインストラクションを，コントローラーに送信する設定にすれば，スイッチが未知のパケットを受け取った際に，逐次 OpenFlow コントローラーに問い合わせを行い，OpenFlow コントローラーでは，該当パケットを処理するための新たなフローエントリーをフローテーブルに書き込むこと

で，新規のパケットの経路制御を実現している．フローテーブルはスイッチ側が保管しているため，OpenFlow コントローラーへは最小限の問い合わせで済み，フローテーブルに基づいた転送処理が行われる．

一般的なスイッチと異なる特徴として，1 つのパケットに複数のフローエントリーを適用することが可能であるという点がある．つまり，正規の宛先に転送しつつ，任意の宛先にも転送することができるため，パケットの柔軟な制御が行える．このようなパケット制御の利点を用いて，取り出したい通信のみを正規の宛先に加えて，コントローラーにも転送することで，可視化端末側に通信が発生したことを通知する仕組みを実装する．

2.1.2 Ryu と Open vSwitch

ユーザーは, OpenFlow コントローラーとして動作するプログラムを自ら作成することによって，OpenFlow ネットワークを構築する．このプログラム部は，全ての実装を自ら行うこともできるが，今回は OSS で提供されている OpenFlow に準拠したフレームワークである，Ryu SDN Framework[7]（以下，Ryu と表記）を OpenFlow コントローラーに採用する．Ryu を採用した理由は以下の通りである．

- 公式の日本語ドキュメントが存在し，理解しやすい
- サンプル実装例及び，サンプルプログラムが充実している
- REST 連携機能を備えているため，OpenFlow で管理されたネットワーク上を流れるパケットやスイッチが保有する情報への参照が容易である

Ryu には REST 連携機能があるため， OpenFlow で定義したネットワーク内のパケット情報を，REST API を経由して配信できる．REST API を利用することで，可視化端末は，特殊な実装を必要とせず，HTTP リクエストによってパケット情報が得られる．そのため，一般的な実装でシステムを構築できると考えた．可視化端末へのパケット発生通知は，リアルタイム性が求められるため, 双方向通信を実現する WebSocket を使用して可視化端末にパケット情報を配信する．

OpenFlow スイッチには, OSS の Open vSwitch を採用する. Open vSwitch は, OpenFlow に準拠した仮想スイッチである．汎用 PC 上で動作し，汎用 PC を OpenFlow スイッチとして運用できる． そのため，OpenFlow に対応したネットワーク専用機器が無くても，汎用 PC があれば手軽に導入することが可能である．

2.2 システム概要

本可視化システムは，図 1 に示す通り，「OpenFlow コントローラー」と「OpenFlow スイッチ」，「可視化アプリケーション」，「Web サーバー」，「クライアント PC」の 5 つのモジュールから構成される．検証環境を表 1 に示す．L2 スイッチとして動作する Raspberry Pi(Switch)には，LAN ポート拡張のためイーサネットアダプターを 2 つ取り付けた．プログラムの動作環境には，他の環境でも問題なく動くことを保証するために，Docker[8]を使用した．Ryu と Web サーバーは，Docker コンテナ内で動作する．

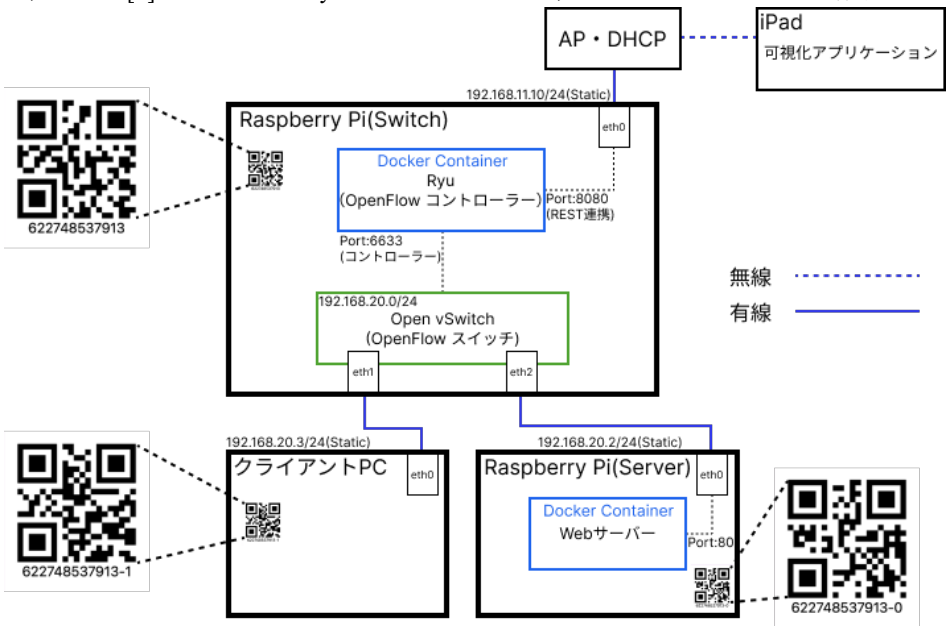

図 1 システム概要図

表 1 検証環境

モジュール	機器	環境
OpenFlow コントローラー, OpenFlow スイッチ	Raspberry Pi 4 Model B	Docker USB Type-A Ethernet アダプター×2
可視化アプリケーション	iPad Pro 11 インチ	
Web サーバー	Raspberry Pi 3 Model B	Docker

各モジュールの役割を説明する．

- OpenFlow コントローラー

  Ryu は，Open vSwitch とのやり取りと REST 連携を使用した iPad とのやり取りでそれぞれ異なる待ち受けポートを使用する．

  Open vSwitch 間では，接続された Open vSwitch に対して，L2 スイッチとして機能させるために，必要なフローエントリーを Open vSwitch のフローテーブルに追加する．また，Open vSwitch が HTTP パケットを受信した際にコントローラーにもパケットを転送するフローエントリーを追加する．

  REST 連携部では，Open vSwitch に接続されている機器の MAC アドレス一覧を返す機能と， Open vSwitch が HTTP パケットを受信した際に，そのパケットのヘッダー情報を WebSocket 通信によって，コネクションが確立された機器に対して配信する機能がある．全て JSON 形式で配信する．

- OpenFlow スイッチ

  Open vSwitch は，Raspberry Pi(Switch)のホスト OS 上で動作する．Raspberry Pi に拡張した eth1 と eth2 を OpenFlow スイッチのポートとして使用する．OpenFlow コントローラーの IP アドレスには，Docker 環境内の Ryu のアドレスを設定する．Ryu から追加されたフローエントリーをもとに転送処理を行う．

- 可視化アプリケーション

  可視化を行う機器には，iPad Pro 11 インチモデルを使用し，実験環境の AP(アクセスポイント)に接続する．可視化アプリケーションは，Ryu と通信する部分と，AR 表示部分の大きく 2 つに分けられる．

  REST API を経由した Ryu との通信では，HTTP 通信によって JSON ファイルを受け取る．

  AR 部分では，Apple 社が提供する ARKit[9]を使用する．マーカーとして使用する QR コードは，REST API 経由で受け取った Open vSwitch に接続されている MAC アドレスの情報をもとに生成する．QR コードを印刷するための機能もアプリケーション上で提供する．AR 機能は，この生成した QR コードを検出対象として動作する．AR で再生するアニメーションでは，WebSocket で受信したヘッダー情報を含んだ JSON ファイルから送信元と送信先の MAC アドレスを参照し，対応する QR コード間で通信の様子を表現する．

- Web サーバー

  適当な HTML ファイルを返す Web サーバー．Web サーバーが動作する Raspberry Pi(Server)の eth0 と Raspberry Pi(Switch)の eth2 を LAN ケーブルで接続する．eth0 には，192.168.20.2 を固定 IP として設定する．

- クライアント PC

  Web サーバーに HTTP リクエストするための任意の HTTP クライアント．用意した任意の PC の LAN ポートと Raspberry Pi(Switch)の eth1 を LAN ケーブルで接続する．PC 側のインターフェースには，192.168.20.3 を固定 IP として設定する．

シーケンス図を図 2 に示す．表示の順序は，以下の通りである．

① Raspberry Pi(Switch)に接続されている機器の MAC アドレスと Open vSwitch に割り当てられた DPID を取得する．

② ①で取得したデータをもとに，QR コードを生成し，iPad の可視化アプリケーションから印刷処理を実行する．

③ 印刷した QR コードの紙をクライアント PC，Raspberry Pi(Switch)，Raspberry Pi(Server)に貼る．

④ Raspberry Pi(Switch)に WebSocket で接続する．

⑤ クライアント PC から Raspberry Pi(Server)の 80 番ポートに HTTP リクエストし，Web コンテンツを受信する．

⑥ 送り先が 80 番ポートのパケットを JSON 形式に変換して，WebSocket で iPad に配信する．

⑦ 送り元が 80 番ポートのパケットを JSON 形式に変換して，WebSocket で iPad に配信する．

iPad 上の AR 表示では，⑥と⑦で取得した情報をもとにしてアニメーションを再生する．

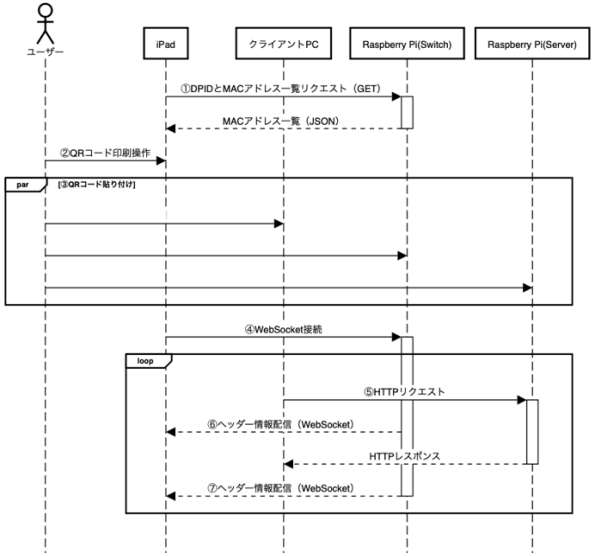

図 2 システムシーケンス図

2.3 主要機能

各モジュールの主要な機能を説明する．

2.3.1 Ryu（コントローラー）

- Open vSwitch を L2 スイッチとして動作させる機能

  Ryu のドキュメントにある，OpenFlow スイッチをスイッチングハブとして利用するためのコードを利用したため，独自に実装した機能はない．

- HTTP パケットをコントローラーに転送させる機能

  Web サーバーは，80 番ポートでリッスンしている想定である．そのため，Web サーバーへアクセスするパケットの TCP ヘッダーは．送信先ポートが 80 番である．また，リクエストに対するレスポンスパケットは，送信元ポートが 80 である．したがって，Open vSwitch を通過する HTTP パケットのみを取り出すためには，TCP ヘッダーの送信先ポートが 80 番であるパケットと送信元ポートが 80 番であるパケットをコントローラーに転送させるフローエントリーが必要となる．これら 2 つのフローエントリーを Open vSwitch に追加する機能を実装する．

2.3.2 Ryu（REST 連携）

- Open vSwitch に接続された機器の MAC アドレスを返す機能

  AR で使用するマーカーを作成するために，Open vSwitch に接続された機器の MAC アドレスが必要である．ここでは，それらを提供するための機能を実装する．Open vSwitch に割り当てられる DPID とスイッチに接続された機器の MAC アドレスの対応関係を図 3 のような JSON 形式で返す．

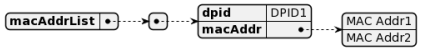

図 3 MAC アドレスリストの JSON 形式

- HTTP パケットのヘッダー情報を配信する機能

  Open vSwitch から送られた HTTP パケットのヘッダー情報を，WebSocket のコネクションが確立された機器に対して図 4 のような JSON 形式に変換してリアルタイムで配信する．

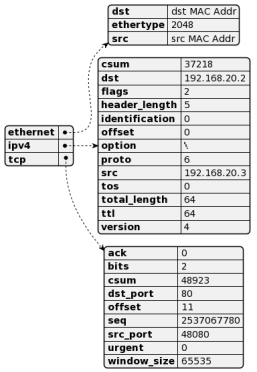

図 4 ヘッダー情報の JSON 形式

2.3.3 可視化アプリケーション

- MAC アドレス一覧を取得する機能

  Open vSwitch に接続された機器の MAC アドレス一覧を Ryu から REST API を経由して，HTTP リクエストにより取得する．

- QR コードを作成する機能

  Ryu から取得した，スイッチの DPID とそれに接続された MAC アドレスをもとに，それらに対応する QR コードを作成する．作成した QR コードは，印刷プレビューから印刷ができる．

- パケット情報を取得する機能

  Ryu との間で WebSocket によるコネクションを確立し，OpenFlow のネットワーク内で HTTP パケットが発生した際に，コネクションを通してリアルタイムでパケットのヘッダー情報を受け取る．

- パケット情報を表示する機能

  Ryu から受信したパケットのヘッダー情報を表示する．

- AR 表示機能

  作成した QR コードをマーカー検知対象として，iPad の外カメラの映像を表示する．パケットのヘッダー情報を受け取ったら，送信元と送信先の MAC アドレスを参照して，対応する QR コード間で 3D の球を動かすアニメーションを再生する．パケットの発信元 MAC アドレスによって球の色を変え， MAC アドレスと球の色の対応関係を，AR 画面の右下に表示する．再生モードに切り替えることで，任意のパケットのみ再生することも可能である．

  アニメーションの様子を図 5 に示す．時系列として，左の図から右へ遷移して

おり，緑色の球が PC から Raspberry Pi(Switch)を経由して，Raspberry Pi(Server)へと移動している様子がわかる．

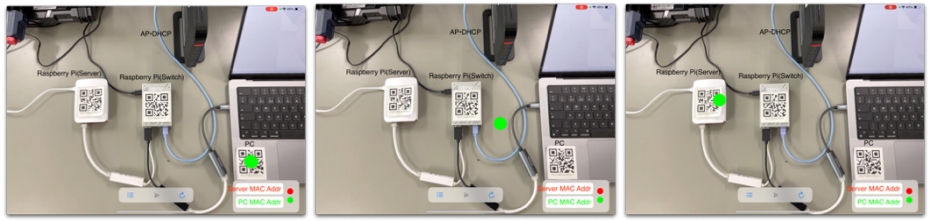

図 5 AR によるアニメーションの様子

2.4 想定する利用例

提案した手法を用いて，高等学校などの教育機関でどのように活用できるかを考える．前提として，本システムは単体で使用されることを想定していない．教科書で学んだ事前知識をもとに補助的に使用されることを想定している．プロトコルに関する知識や，ネットワークの仕組みを学んだ上で，生徒が手を動かして実際に発生した通信の動作を本システムで確認することで，動作原理の理解が深まるのではないかと考える．慣れ親しんだ操作で発生させたパケットの様子が，AR 上で可視化されることにより，実機の PC を操作することによる親近感と AR で表現することによる想像のしやすさの両方の長所を生かした体験を提供できる．このような体験は，擬似的もしくは仮想的な通信による説明では得難い，大きな違いである．

3. 結論

3.1 検証と考察

研究目的である，導入のしやすさと環境の再現性に関しては，次の 2 つの観点で，おおよそ達成できたと考える．

1 つ目は，必要とする機材の調達コストである．本研究では，Raspberry Pi を用いてスイッチや Web サーバーを動作させたが，Raspberry Pi 以外の他の汎用 PC で代替できる．そのため，多くの学校に導入されている PC を使用して同様の環境を用意することができ，追加の費用をできるだけ抑えることができる．このシステムを構成する上で，汎用 PC 以外に必要となるものは，イーサネット変換アダプタが挙げられる．多くの汎用 PC は，標準で 1 つの LAN ポートしか搭載していないため，必要に応じて変換アダプタでポートを拡張する必要がある．しかし，ネットワーク専用機器を導入するよりはコストを抑えられるため，妥当的な構成ではないかと考える．

2 つ目は，OpenFlow や Docker を使用することによる環境整備のしやすさである．OpenFlow は，コントローラーによってネットワーク構成を定義するため，今回使用した Ryu のプログラムを実行するだけで，本研究で使用したネットワークを構築できる．Ryu や Web サーバーは，Docker コンテナ内で動作しているため，ホスト OS に Docker がインストールされている環境であれば，いくつかの Docker コマンドだけで機器の差異に影響されずに，環境構築することが可能である．しかし，現状では，これらのコマンドに関する，ある程度の専門知識を有していることが前提となっている．

パケット通信の様子を AR で表現することにより，実際に発生したパケットをインタラクティブに表現することが達成できた．実物の機器を表示しつつ，パケット通信をアニメーションで表現することで，利用者は直感的で親近感のある体験ができると考える．

検証では，研究室の学生 7 名にシステムを操作してもらい，以下のフィードバックを得た．

- ある程度基礎知識のある人が使えば，理解を深められるシステムであると思う
- 講習会や授業などで補助的に使うことができれば理解が深まると思う
- 3 ウェイハンドシェイクが可視化されていて，わかりやすい
- 他のプロトコルとの違いが見られるともっとわかりやすい
- アニメーションが早すぎて視線で追うのが難しい
- 実際の通信の速さがあると参考になる

HTTP 通信の 3 ウェイハンドシェイクの様子が可視化されることでわかりやすいという本研究が意図する好意的な意見もあったが，高等学校で使われるためには，解決すべき課題がいくつか残されていることがわかった．

3.2 今後の展開

顕在化した課題について今後の展開を検討する．

本研究では，通信の例として HTTP 通信を取り上げた．これに加えて，UDP 通信を用いる DNS などのプロトコルと比較することで，プロトコルごとの違いを従来よりも分かりやすく表現することができると考える．また，アニメーションの表現方法に関しても，本研究では実装に至らなかったが，QR コード間を直線的に移動するのではなく，接続された LAN ケーブルに沿って移動するようにしたり，機器がパケットを送受信するときに，階層モデルの層ごとにカプセル化される様子を AR で表現したりすることで，目標を達成するシステムに一歩近づくと考える．

一方で，導入面でも課題が残されている．教育現場で幅広く使われることを考慮すれば，専門知識がなくても簡単にセットアップできることは必須の条件である．GUI で容易にセットアップができるようにするなど，依然として導入のハードルを下げるための改善の余地がある．

3.3 おわりに

本研究では，導入のしやすさを考慮したネットワーク通信の新たな表現方法を提案した．表現方法に AR を取り入れることで，機器を目の前にしながらその内部の動作を仮想的に表現した．検証によって，教育現場で使われるために，いくつかの課題が浮き彫りになった．学習者がより直感的に理解できるシステムを目指して，引き続き検証していきたい．

参考文献

[1]文部科学省：GIGA スクール 構想の実現へ，入手先〈https://www.mext.go.jp/content/20200625-mxt_syoto01-000003278_1.pdf〉（参照 2024-02-13）．
[2]文部科学省：義務教育段階における 1 人 1 台端末の整備状況（令和 4 年度末時点），入手先〈https://www.mext.go.jp/content/20230711-mxt_shuukyo01-000009827_01.pdf〉（参照 2024-02-13）．
[3]文部科学省：高等学校学習指導要領（平成 30 年告示），入手先〈https://www.mext.go.jp/content/20220324-mxt_kouhou02-000021499_1.pdf〉（参照 2024-02-13）．
[4]石川有彩，吉原和明，井口信和，渡辺健次 : OpenFlow を用いたネットワーク学習教材の開発，研究報告インターネットと運用技術（IOT），Vol.2021-IOT-52，No.2，pp.1-7（2021）．
[5]Open Networking Foundation : OpenFlow Switch Specification Version 1.5.1 (Protocol version 0x06), 入手先〈https://opennetworking.org/wp-content/uploads/2014/10/openflow-switch-v1.5.1.pdf〉（参照 2024-02-13）．
[6]Linux Foundation：Open vSwitch，https://www.openvswitch.org（参照 2024-02-13）．
[7]Ryu SDN Framework Community：Ryu SDN Framework，https://ryu-sdn.org/index.html（参照 2024-02-13）．
[8]Docker, Inc.：docker，https://www.docker.com/ja-jp/（参照 2024-02-13）．
[9]Apple Inc.：ARKit，https://developer.apple.com/documentation/arkit/（参照 2024-02-13）．