2023 年度 秋学期

卒　業　論　文

リアルワールドメタバースの実現に向けた動的環境の VR 空間への反映手法

指導教員：柴田 史久

立命館大学　情報理工学部

卒業研究 3 (DB)

コース　：　実世界情報

学籍番号：　2600200173-9

氏　　名：　小林 洋輝

概要

本論文では，現実世界での体験をより豊かにするためのコンセプトとして，リアルワールドメタバースを基盤とした MR キャンパスを提案している．これは従来のメタバースと違い，現実を起点として，現実世界に焦点を当てており，VR/MR の両空間をデジタルツインとして構築することで，互いの空間で相互にさまざまな作業ができるものを想定している．

VR/MR 両空間でのデジタルツインの構築のためには，両空間の三次元空間を一致させる必要がある．しかし，実空間の動的環境を VR 空間に反映するためには，なるべく高い応答性を必要とするものの，モデリングや三次元再構成などでは非現実的であるといった課題がある．ここで，事前に VR 空間に必要と考えられる物体のみをリストアップし，それらの物体のモデルを事前に作成した上で，実空間を定期的にセンシングし，それらの物体の位置姿勢や大きさを取得することによって，VR 空間に必要な物体のみを反映するシステムの提案を行い，簡易的に試作した結果について述べる．

対象とするセンシングデータには，実空間を定期的に周回する RGBD カメラからの画像データおよび深度データが含まれる．本論文で示すシステムの用途としては，リアルワールドメタバースを題材とした VR/MR 環境において，動的環境を高頻度で VR 空間に反映することで，実空間と VR 空間との間で矛盾をなるべく減らすことを目的とする．

本研究では，検討を行なったシステムおよび通信の構成に基づいたものを試作し，実際に通信の遅延や通信量，VR 空間での位置や向きのずれについて実験することで，本システムの可用性を検証した．通信の遅延及び通信量に関しては改善点があるものの良い結果が得られた一方，三次元位置姿勢の計算に関しては誤差による問題が目立つ結果となった．

キーワード

メタバース

リアルワールドメタバース

VR

AR

MR

デジタルツイン

物体検出アルゴリズム

目 次

第1章 はじめに

近年，バーチャルリアリティ（Virtual Reality; VR）は急速に進化し，日常生活や様々な分野において不可欠な技術となっている．バーチャルリアリティとは，コンピュータを用いて構築された，現実世界とは異なるバーチャル環境（VR 環境）にユーザが没入する技術である．この技術は視覚や聴覚，触覚などあらゆる感覚を活用し，ユーザに現実世界と区別がつかないほどの感覚を提供する．

理想的なバーチャルリアリティが満たすべき基本的な特徴は**図** 1.1 であらわされる[1]. 実時間相互作用とは VR 環境と現実世界がリアルタイムに相互作用をしながら自由に行動をできることを表す．これによってユーザは VR 空間内での体験にリアルタイムで対応でき，より没入感のある体験を得ることができる．三次元空間はユーザにとって違和感のない三次元空間を構成していることを表す．これによって，ユーザは視覚的に自然な情報を得ることができ，よりリアルな体験が可能となる．自己投射はユーザが VR 空間に入り込み，シームレスにつながっていることを表す．これによりユーザは VR 空間内での存在感を実感し，自信を具現化できることを指す．例えば，3DCG によるバーチャルライブでは，自然な三次元空間およびその場に居るという自己投射感覚はあるがユーザとの相互作用はないため，理想的なバーチャルリアリティとは言えない．

一方，バーチャルリアリティの進化とともに，新たな概念としてメタバース（Metaverse）が台頭している．これはギリシア語で「超える」を意味する接頭語であるメタ（Meta）に「宇宙」や「世界」を意味する単語であるユニバース（Universe）を有報した合成語であり，主にセカンドライフのような三次元 VR 空間を総称してメタバースという言葉が使われている．ここで，バーチャルリアリティ学においては，次の条件を満たすものを特にメタバースであると定義されている[1].

- 三次元 VR 空間を持つ

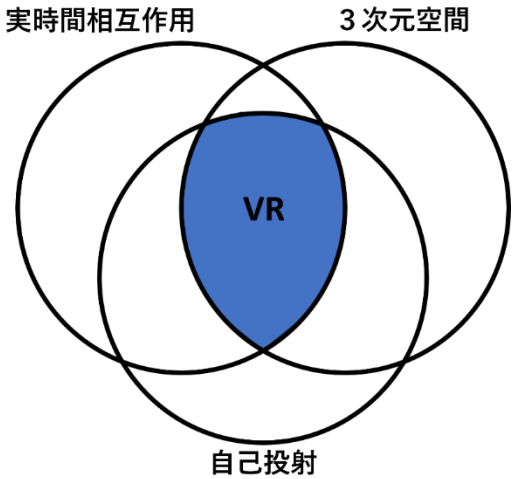


図 1.1　VR の三要素

- 自己投射性を満たすアバタが存在する
- 複数のユーザが同一 VR 空間を共有できる
- VR 空間内に，オブジェクトを創造できる

メタバースが三次元 VR 空間を持つことは，ユーザが現実世界と同じほどに立体的かつ臨場感のある環境に没入できることを指し，これによって従来の二次元インタフェースを超えてより自然な体験が可能となる．また，自己投射性を満たすアバタの存在および複数のユーザが同一 VR 空間を共有できること，VR 空間内にオブジェクトを創造できることは，ユーザが VR 空間において自信を具現化し，ほかのユーザとコミュニケーションをとることができることを示し，これはリアルタイムにそれぞれのユーザによる様々なバーチャルコミュニケーションの可能性を提供する．

以上の定義に加えて，メタバース進化論ではさらに以下の条件を満たすものをメタバースとしている[2].

- ユーザ間でコンテンツやサービス，金銭の交換が可能
- 目的別に使用デバイスを選択できる
- 参加手段として AR/VR などがある

これらの条件から，メタバースは VR に限らず，様々なデバイスによる様々な参加手段が存在し，また資産の交換ができる．以上から，メタバースは理想的なバーチャルリアリティが満たすべき三要素をすべて満たしているといえる．

メタバースを題材とした作品の事例として，マトリックス（The Matrix）やソード・アート・オンラインといったものがある．どの作品においても，現実世界において指定のデバイスを装着することによって，自身のアバタに憑依して三次元 VR 空間に参加しており，メタバース技術が発達した未来世界が描写されている．

メタバースサービスの代表的なプラットフォームとして，VRChat や Cluster といったものが知られている．これらはどれも三次元 VR 空間においてユーザがアバタのカスタマイズおよびリアルタイムにほかのユーザとコミュニケーションをとることができるプラットフォームであり，メタバースの自己投射性や複数ユーザとの空間共有といった要素を満たしている．テキストチャットや音声のみのコミュニケーションとの大きな違いとして，従来のコミュニケーション方法に加えてアバタの身振り手振りなど多種多様な表現方法を用いた，より自由度の高いコミュニケーションを行うことができる．

また，3D オンラインゲームである Fortnite やあつまれどうぶつの森など，ゲーム本来の目的を超えてコミュニケーションツールとして，メタバースの代表として扱われることもある[3]．これらのゲームは，単なる娯楽の枠を超え，ユーザたちが VR 空間内で交流し，アバタを通じたコミュニケーションを深める場として機能している．これは，メタバースが単なる VR 空間を超え，現実社会と密接に結びつく可能性を示している．

他に，メタバースを構成する三次元 VR 空間として，現実世界の地理情報などの情報を再現したものを扱うことがあり，これをデジタルツイン（Digital Twin）と呼ぶ．デジタルツインを利

用したメタバースの代表的なものとしては東京大学メタバース工学部というものが知られている．これは文部科学省によって正式に承認された大学の一部ではなく，工学部生に限らず中高生や社会人，学生など様々な層に工学分野に関する情報を普及させるための新たな教育プラットフォームである[4].

近年バーチャルリアリティ技術の向上によって，現実との区別がつかないほどの没入感や臨場感を提供している．これに伴ってメタバースの基盤および国内市場規模は指数関数的に急速に成長しており，2022 年度時点で 1337 億円であるが，2027 年には約 2 兆円にも達すると予測されている[5]（**図 1.2**）．これはメタバースが単なるエンターテインメントの枠を超えてさまざまな分野において拡大し，人々の日常生活や教育，ビジネスにおいて不可欠な存在となる可能性を示唆している．

一方で，全てがデジタルで構築される VR とは異なり，現実世界を起点としてバーチャルな情報を現実世界に重畳描画するものである拡張現実感（Augmented Reality; AR）や，現実世界と VR 世界を融合させたものである複合現実感（Mixed Reality; MR）に関する研究が注目されている[6][7]．AR/MR は現実を起点としているといった特徴から，健康，医療，スポーツ，福祉，芸術，娯楽などの幅広い分野での応用が期待されている．

例えば，AR アプリケーションとしては，スマートフォンのカメラを利用して画面上で家具配置のシミュレーションを実現できるものが存在し，MR は研修といったトレーニングの現場で幅広く活用されており，体験者の作業に合わせて解説や操作方法を現実世界に重畳描画することで視覚的に理解しやすく作業の効率を向上させることができる．

ここで，VR で構成されていることが多いとされる従来のメタバースに，AR/MR と融合させたリアルワールドメタバースというものが存在する．筆者が所属している研究室では，このリア

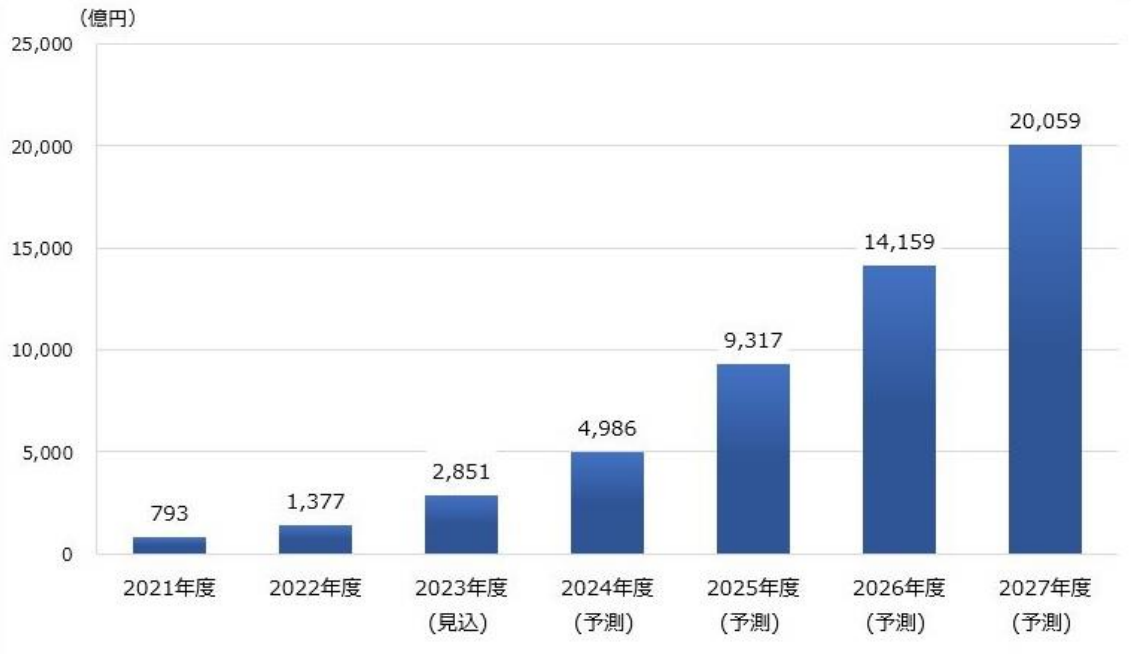


図 1.2　メタバースの国内市場規模推移・予測［5］

ルワールドメタバースというコンセプトを具現化する一環として, MR キャンパスというものについて提案をし，研究を進めている．

本論文の構成は以下のとおりである．

第 2 章では，リアルワールドメタバースについて，その概要について述べたのち，関連研究について述べる．そして，さらに先に述べた MR キャンパスについてその概要および課題について述べたのち，本研究の方針について述べる．

第 3 章では，第 2 章で述べた本研究の方針より，現実世界を VR 空間にリアルタイムで反映するデジタルツイン構築システムの開発について述べる．

第 4 章では，第 3 章で述べたシステムについて，実際に試作することによってその性能について確認する．

最後に，第 5 章では，本論文をまとめ，今後の展望について述べる．

第2章　リアルワールドメタバース

本章では，筆者の所属する研究室で研究中のリアルワールドメタバースについて，その概要や必要性，特徴について述べる．

2.1　リアルワールドメタバースの概要

近年，VR およびメタバースの急速な進化に伴って，メタバースと AR/MR が融合する新たな形態が生まれ，これを従来の VR を中心としたメタバースと区別してリアルワールドメタバースと呼ぶ．これは Niantic 社が提唱したものであり，従来のメタバースが VR 空間のみを用いて体験できるのに対し，リアルワールドメタバースは実空間のデジタルツインを構築し，実空間の物理的要素と AR/MR のデジタル情報を密接に結びつけることによって現実を起点として現実の体験をより豊かにすることを目的としている．

2.2　関連研究

リモートユーザが実際に実空間に存在しているかのような高い臨場感と存在感を持ってコミュニケーションを行うための技術をテレイグジスタンス（Telexistance）という[1]．臨場感とは，VR 空間においてユーザが実際にその場に居るような感覚のことで，視覚や聴覚，触覚，嗅覚，味覚，前庭感覚などすべてがそろうことで理想的である．存在感とは，他者ユーザから見て，あるユーザが確かにその空間にいると感じられる感覚のことである．

現実を基盤とした AR/MR アプリケーションとして，MRMAC（Mixed Reality Multi-user Asymmetric Collaboration）というものがある[8]．これは，360 度カメラを使用して実環境を VR 空間にリアルタイムでストリーミングを行うことで，実空間のデジタルツインを VR 空間にリアルタイムに構築している．リモートユーザはヘッドマウントディスプレイ（Head Mounted Display; HMD）などの AR/MR デバイスを用いることでこの空間に仮想的にテレポートすることができ，ここにアバタやリモートユーザのホログラム，オーディオなどのリアルタイムなストリーミングを組み合わせることでより高い臨場感および存在感を提供する．これによって，従来のビデオ会議よりも優れた，現実世界のユーザとリモートユーザとのコミュニケーション，強いてはテレイグジスタンスを実現する．

ほかに，AR/MR を活用したアプリケーションには Dynamics 365 Remote Assist や Vuforia Chalk がある[9][10]．これらはどちらも現実世界のユーザとリモートユーザが共同作業を行うことを支援するためのアプリケーションである．例えば Dynamics 365 Remote Assist は，現実世界のユーザが HoloLens などのスマートデバイスを使用することで映像をリモートユーザと共有し，専門家や技術者が実際にその場に居なくても，リモートユーザとして現実世界のオブジェクトにバーチャルな注釈などを入れることで，リアルタイムにその場に居るユーザに指示を送り問題解決の支援を行うことができる．

2.3 MR キャンパス

先に述べたように，筆者の所属する研究室では，リアルワールドメタバースのコンセプトを具現化する一環として，「MR キャンパス」というものについて提案をし，研究を進めている．まず初めに，MR キャンパスにおける「キャンパス」とはあくまでリアルワールドメタバースのコンセプトを具現化するにあたって大学のキャンパスを想定しているだけであり，本質的にはキャンパスに囚われずあらゆる場所において実現できるということを強調しておく．

まず，従来の VR メタバースでは VR 空間においてユーザが対等にコミュニケーションを行えるため臨場感，存在感がともに満たされているが，実際に実空間に影響を及ぼすことはない．一方，リアルワールドメタバースのコンセプトでは現実を基準としたうえで，その場に居ないリモートユーザとの共同作業を行うことよって問題解決を円滑に行えるが，リモートユーザの存在感および臨場感を満たす表現方法については課題が残っている．

MR キャンパスは，リアルワールドメタバースに従来の VR メタバースを組み合わせることによってリアルワールドメタバースにおけるリモートユーザのテレイグジスタンスを実現することを目的にしている．

MR キャンパスの構想を**図 2.1** に示す．構想しているアプリケーションでは，キャンパスに実際にいるユーザとその場に実際にはいないユーザによって，アクセスの種類が異なる．ここで，本論文において，前者を実空間側，後者をリモート側と呼ぶこととする．実空間側は現実であるキャンパスを起点とした MR 空間に参加し，リモート側はキャンパスを元にした VR 空間に参加する．なお，VR 空間は MR 空間のデジタルツインであり，実空間の環境変化（動的環境）が反映される．リモート側のユーザは従来のメタバースと同様に VR デバイスや PC などを用いることによって自身のアバタに憑依して VR 空間に参加するが，同時に VR 空間におけるアバタの位置姿勢および音声を MR 空間と同期して反映する．同様に，実空間側のユーザは MR デバイスやスマートフォンなどを用いて MR 空間に入り，ユーザの実空間上における位置姿勢や音声を VR 空間上の自身のアバタと同期して反映する．また，他に，リモート側のユーザの参加方法としては，実空間に存在するロボットに憑依することで VR 空間ではなく，MR 空間に参加するといったアクセス方法なども構想している．

この構想では，実際にキャンパスにいる実空間側のユーザと実際にはその場にいないリモート側のユーザという異なるユーザグループが異なるアクセス方法を用いて，現実を起点とした MR 空間とそのデジタルツインである VR 空間を介することによって，両空間のユーザが互いに交わって対等に多種多様なコミュニケーションを取ることができる．例えば，教育分野などにおいて新たな学習体験を提供できる可能性がある．例をあげると，国境などの地理的な制約を超えて異なる場所にいるユーザが VR 空間から実空間側と同じ授業に参加し，質問やディスカッションなどを，その場にいるかのような感覚を持ちより効果的に共同学習を行うことができると考えられる．また，産業界などにおいても活躍が期待される．例えば，異なる場所にいるエンジニアおよびデザイナーが，実空間と VR 空間を介してクリエイティブなアイデアの共有や問題解決を行う事ができると考えられる．

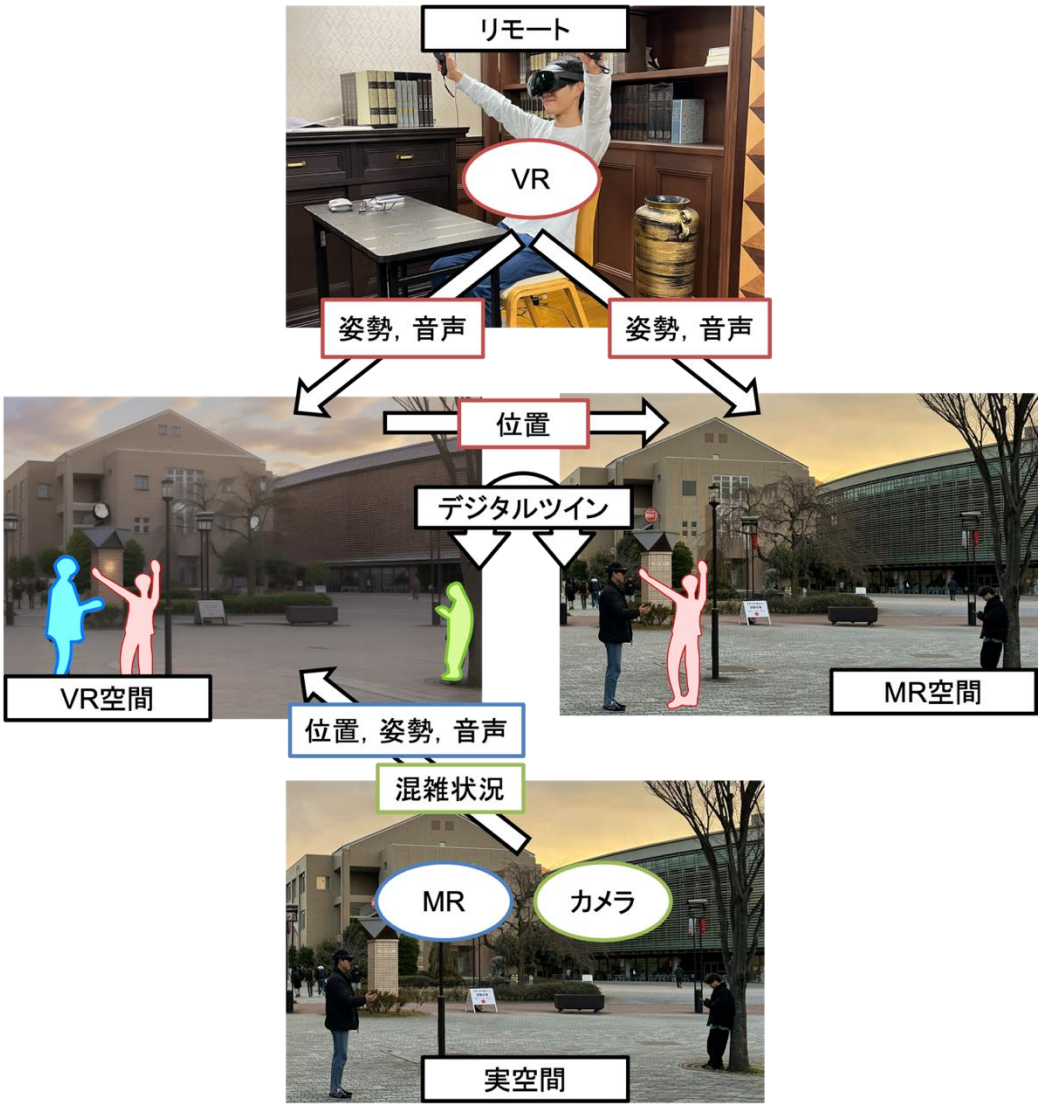


図 2.1　MR キャンパス構想図

2.4　本研究の位置づけ

MR キャンパスの実現における最優先課題を大きくまとめると以下のようになる．

- リモート側

(A-1) 位置姿勢の VR/MR 両環境のアバタへの反映

(A-2) 音声の VR/MR 両環境への反映

- 実空間側

(B-1) 位置姿勢の VR 空間のアバタへの反映

(B-2) 音声の VR 空間のアバタへの反映

(B-3) 動的環境の VR 空間への反映

(B-4) 混雑状況の VR 空間への可視化

リモート側のユーザは従来のメタバース同様に VR 空間に参加するが，この時にユーザの操

作によって VR 空間における位置姿勢を変更し，同時にデジタルツインである MR 空間に反映する必要がある（A-1）．また，ユーザの音声も同様に VR 空間における位置姿勢を考慮した位置に反映するだけでなく，MR 空間にも反映する必要がある（A-2）．

実空間側は基本的に MR 空間に対して特定の行動を行うことが必須ではないため，VR 空間への反映が主な課題となる．まず，実空間側ユーザの位置姿勢を VR 空間のアバタへ反映するためには，実空間側においてどの場所にユーザがどんな姿勢で存在するかをセンシングする必要がある（B-1）．次に，両空間でコミュニケーションを行うために，実空間側のユーザの音声は VR 空間においてユーザのアバタの位置から発するようにする必要がある（B-2）．そして，動的環境を VR 空間に反映する必要がある（B-3）．これは，仮に事前に実空間をすべてモデリングしたものを VR 空間とした場合，実空間の環境変化によって VR 空間が変化しないことになり，実空間のデジタルツインとしての意味を失う．この場合，仮に実空間に実際に存在しているあるオブジェクトを移動させた際においても VR 空間には変化が起きない．これによって，そのオブジェクトが元々置いてあった場所に実空間側のユーザが立っていたときに，オブジェクトとユーザのアバタが同じ座標に存在することになり，リモートユーザからは実空間側のユーザのアバタがオブジェクトに埋まってしまうといった矛盾が生じる．そのため，実空間のデジタルツインの構築をする上で，動的環境を VR 空間へ反映する必要性がある．そして，デジタルツインを構築するうえで，実空間の混雑状況に関してリモートユーザは実空間ユーザと同様に確認できるようにするべきである（B-4）．

本論文では，これらの課題のうち，実空間のデジタルツインの構築に関して動的環境の VR 空間への反映に関して解決を目指す．

第3章 デジタルツイン構築システムの試作

本章では，デジタルツイン構築に関する前提条件について述べたのち，関連した研究や技術について述べ，本論文で提案した MR キャンパスにおいて必要となるデジタルツイン構築システムの試作内容について述べる．

3.1 前提条件

MR キャンパスを実現するうえで実空間のデジタルツインの構築方法には大きく分けて 3 通りある．1 つ目は実空間を三次元再構成することであり，2 つ目は実空間になるべく似せた VR 空間をモデリングによって再現すること，3 つ目は物体検出を用いて必要な物体のみを VR 空間に事前に用意した 3D モデルを配置することである．ここで，構築された VR 空間はデジタルツインであるために，実空間との矛盾を起こすべきでない．主な課題として，1 つ目は実空間に存在する人間を除去する必要があり，2 つ目は動的環境に対応する必要がある．その理由としては，前者は本システムに参加するユーザが三次元再構成された状態で VR 環境にアバタが重畳描画されてしまい，後者は前章で述べたとおりである．したがって，2 つ目の案は棄却される．また，3 つ目は現状検出アルゴリズムが完全でないことや，事前に 3D モデルを何パターン用意すればいいのかという妥協点を決める必要がある．

3.2 関連研究・技術

実空間をリアルタイムに三次元再構成を行い VR 空間に反映するシステムとして，Remixed Reality というものがある[11]．このシステムでは，Kinect v2 を 8 台環境に設置することで，$4m \times 5m$の部屋をリアルタイム三次元再構成し，VR 空間に反映することが可能になる．さらに，三次元再構成されたモデルにおいて，物体の任意な移動，複製，除去が可能である．

また，事前に用意した三次元モデルを利用して三次元物体検出を行うシステムとして，NeRF-RPN というものがある[12]．このシステムでは，物体の位置や大きさ，向き，種類などの情報を取得できるため，任意の物体のみの除去が可能である．

現在，インターネットにおいて広く利用されているデータ通信プロトコルとして TCP/IP がある．これは Transmission Control Protocol (TCP) と Internet Protocol (IP) の組み合わせで構成され，TCP は通信相手の確認手順を踏むことでデータの信頼性を高め，エラー制御や再送制御などの機能を提供するコネクション指向のプロトコルである．

TCP/IP と対照的に，データ通信プロトコルとして User Datagram Protocol (UDP)がある．UDP は TCP と異なり，データの信頼性を向上させるための処理を含まず，信頼性を追求しないコネクションロス型のプロトコルである．そのため，UDP は処理やパケットの構造が簡素であり，リアルタイム性を重視した通信において利用される．

TCP/IP および UDP に加えて，データ通信プロトコルの選択肢として Data Distribution Service (DDS) がある．DDS は分散システムにおいてデータ通信を容易にするための標準的な

プロトコルおよびAPIを提供しており，TCPやUDPが主にインターネット上での通信に焦点を当てているのに対し，DDSはリアルタイムおよび優先度が高いシステムでのデータ通信に特化している．TCP/IPが信頼性を重視し，UDPが簡素かつリアルタイム性を追求する中，DDSはリアルタイム制と柔軟性を両立させ，分散環境でのデータ通信に高い効率性をもたらす．

ロボット分野やITS分野において，開発と制御を支援するためのオープンソースソフトウェアとしてROS（Robot Operating System）を用いて通信部分の実装を行うことがあり，筆者の所属する研究室においても歩行者飛び出し予測の周辺車両の透過システムの開発でこのROSを利用したシステムの拡張を行なっている[13].

ROSはロボット開発を効率的に進めるにあたって有用なソフトウェアプラットフォームであり，その機能の一部として，分散システムの構築に重要な役割を果たすノード間通信を提供しており，複数の端末間でデータをやりとりする仕組みを提供している．各ノードは独立してリアルタイムにデータを処理する非同期通信であるため，各ノードの通信頻度によらずそれぞれが同時に作業を進めることができる．

ROSにおいて提供されるトピック通信は出版-購読型モデル（publish-subscribe model; pub/sub）に基づいており，従来の通信モデルと異なり送信側と受信側で特定の相手を設定する必要がない．送信者側はデータに特定の識別子をつけた上でネットワークへ送信し，受信側はこの識別子を基にして受信したいデータを受け取る．

ここで，ROSにはROS1とROS2が存在する．主な違いとしては，ROS1はTCP通信であるため，通信の確実性が高まる代わりに通信のオーバヘッドが発生し，遅延が生じる可能性が高まり，リアルタイム性が確保できない可能性があるのに対し，ROS2ではDDSを使用しているため，リアルタイム性が重要であり一部のデータ損失が許容される場合などに対応することができる．

Unityにおいて ROSを利用するためのライブラリとして，Unity-Robotics-Hubというものがある[14]．これはUnity公式から配布されているライブラリで，TCPベースの通信を行う上で，ROSとUnity間で通信を行うために単一障害点を立てる必要がある．単一障害点とは，システムやプロセスにおいて，その要素が全体の機能において不可欠であり，それが故障した場合に全体が影響を受ける点のことである．

3.3　実装概要

3.2節で述べたRemixed Realityは任意のオブジェクトの除去が可能だが，キャンパス単位で実行しようとするとKinect v2を多く要し，また通信は学内Wi-Fiを想定しているため，通信量は全体で数 10MB/s 程度が望ましいが，キャンパス単位で三次元再構成されたモデルをリアルタイムで通信する場合，通信量は毎秒ギガバイト単位で必要であると考えられ，リアルタイム三次元再構成の方針はMRキャンパスに向きではない．また，NeRF-RPNでは，例で使われている三次元モデルが完全な3Dモデルだが，実際に実空間をすべてセンシングしたうえで三次元再構成するには，人の手を必要とした上で数日程度要すると考えられる．したがって，これも今回

の実装に不向きであると考えられる．

そのため，本論文では，事前に 3D モデルをいくつか用意しておき，二次元物体検出を行ったのちに深度データと合わせて三次元位置を算出し，該当する 3D モデルを動的に VR 空間に配置するという方針でシステムを開発する．

具体的には，事前に壁や床といった半永久的に移動することのないオブジェクトのみをモデリングしておき，その他の机や椅子といった移動性のあるオブジェクトは動的に物体検出を用いて座標を取得し，該当する 3D モデルを配置する．ここで，物体検出のセンシング方法としては，キャンパスを周回するロボットにセンサを乗せることを想定している．

図 3.1 に今回実装したシステムの処理手順について示す．まず，RGBD (RGB, Depth) カメラから画像データを物体検出部に送信し，検出物体の画像内における二次元位置や大きさを取得し，物体位置姿勢計算部に送信する．同時に，深度データとカメラの位置姿勢を物体位置姿勢計算部に送信し，先ほど取得したデータを合わせて，物体の二次元位置における深度から物体の三次元位置を算出し，モデル配置部にこれまでのデータをまとめて送信する．

3.4 実装詳細

本節では，図 3.1 および次に示す**図 3.2** を参考に，今回実装したシステムの詳細について順を追って述べる．

3.4.1 通信手法

3.2 節で述べたように，ROS2 はロボット開発に有用なオープンソースソフトウェアであり，様々なハードウェアやソフトウェアにおいてリアルタイム通信および分散型のシステムをサポートした API を提供しているため，通信に関して詳細な実装をすることなく，アプリケーションの開発に専念することができる．今回は大まかなシステムの実装を主目的としているため，主な通信手法として ROS2 を採用した．

3.4.2 RGBD データ送信部

画像データおよび深度データ（RGBD データ）を取得したのち，物体検出部に画像データを送信すると同時に，物体位置姿勢検出計算部に深度データと RGBD カメラ自身の実空間における位置姿勢（x=cx, y=cy, z=cz, pitch, yaw, roll），およびカメラの有効深度や水平視野角，垂直視

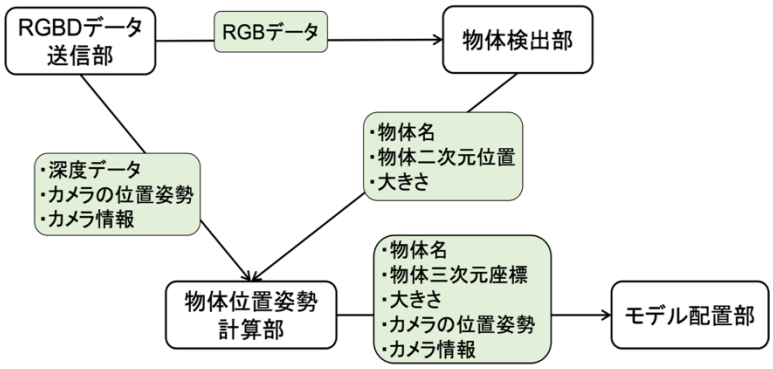


図 3.1　3D モデル動的配置システムの処理手順

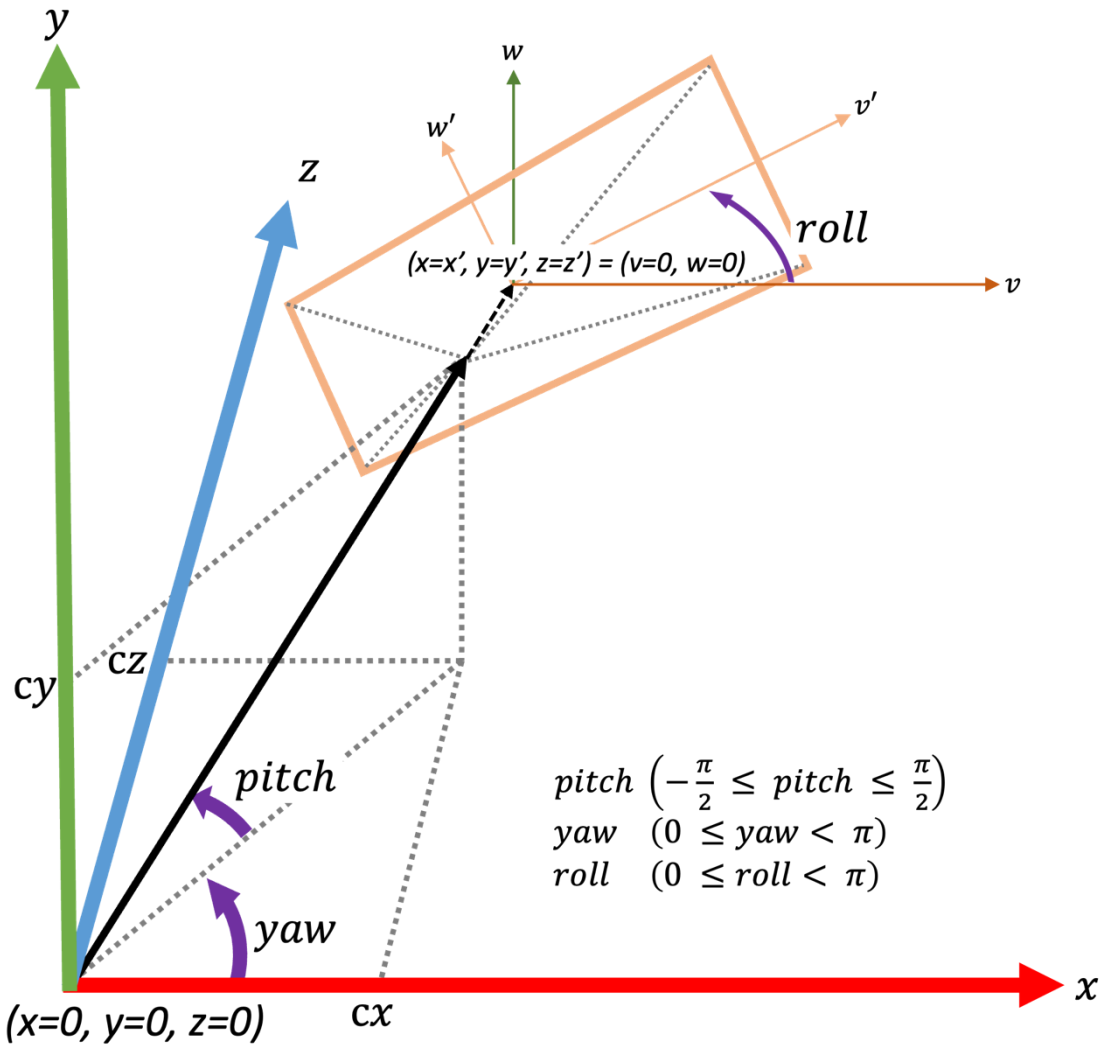


図 3.2　システム内における位置姿勢情報の座標系

野角といったカメラ情報を送信する．ここで，図 3.2 に示されている位置情報の座標軸は Unity 上の座標軸に合わせている．姿勢情報については，pitch は x-z 平面と y 軸間の角度を表し y 軸の正の向きに近いほど大きくなる．次に yaw は x-z 平面上における角度を表し，x 軸から半時計周りに大きくなる．これまでの姿勢情報から，カメラのある深度における視野平面を考え視野平面の横軸を v'軸，縦軸を w'軸とする．ここで，x-z 平面と並行な軸を v 軸とし，v 軸を半時計周りに 90 度回転させた軸を w 軸とする．この時の v-w 平面からの v'-w'平面までの回転角度を roll とする．ここで，オイラー角にはジンバルロックという，特定の姿勢においてオイラー角が一つに定まらないという問題が存在する．今回の場合においては，

$$pitch = -\frac{\pi}{2},\ \frac{\pi}{2} \tag{3.1}$$

となる際，x-z 平面が視野平面と並行になるため．yaw と roll の片方が任意の値を取ることができる．ここで，今回は姿勢を pitch, yaw, roll の順で決め，より姿勢に近くなるように決定した．

3.4.3　物体検出部

RGBD データ送信部から受け取った二次元画像データから，二次元物体検出アルゴリズムを用いて物体を検出し，物体名と物体の二次元画像におけるピクセル位置，物体の高さと幅の二次元のピクセル単位の大きさを物体位置姿勢計算部に送信する．

3.4.4　物体の三次元座標計算手法

RGBD データ送信部および物体検出部からデータを受け取り，まず物体の二次元画像データ内におけるピクセル位置 p および画像の解像度 r を取得し，二次元画像中心から画像端までの距離を 1m に正規化した時の画像中心から物体までの画像における距離 lを以下の計算式で求める．

$$l = \frac{\mathrm{p} - \frac{\mathrm{r}}{2}}{\frac{\mathrm{r}}{2}} \tag{3.2}$$

次に，カメラの視野角θより，画像サイズが正規化した時の大きさになるようなカメラから二次元画像中心までの深度 dを以下の計算式で求める．

$$d = \frac{1}{\arctan\left(\frac{\theta}{2}\right)} \tag{3.3}$$

式 (3.2)および式 (3.3) より画像中心から物体中心までの角度φを以下の計算式で求める．

$$\varphi = \arctan\left(\frac{l}{d}\right) \tag{3.4}$$

式 (3.4) と二次元画像データにおける物体のピクセル位置における深度 d'より，カメラから物体位置における v'-w'平面の原点までの深度 d''を以下の計算式で求める．

$$d'' = d' \times \cos(\varphi) \tag{3.5}$$

また，物体のピクセル単位の大きさ sを，以下の計算式で正規化する．これを s'と置く．

$$s' = \frac{s}{\frac{r}{2}} \tag{3.6}$$

式 (3.2)，式 (3.3)，式 (3.5)，式 (3.6) より，比を用いて画像中心から物体までの実際の距離（v'-w'平面における座標）および，実際の物体の大きさ s''を以下の計算式で求める．

$$\begin{bmatrix} v' \\ w' \end{bmatrix} = l \cdot \frac{d''}{d} \tag{3.7}$$

$$s'' = s' \cdot \frac{d''}{d} \tag{3.8}$$

次に，式 (3.5) およびカメラの三次元位置 (x, y, z) および姿勢 (yaw, pitch, roll) を用いて，画像中心の三次元座標 (cx, cy, cz) を以下の計算式で求める．

$$\begin{bmatrix} cx \\ cy \\ cz \end{bmatrix} = d'' \cdot \begin{bmatrix} \cos(pitch) \cdot \cos(yaw) \\ \sin(pitch) \\ \cos(pitch) \cdot \sin(yaw) \end{bmatrix} + \begin{bmatrix} x \\ y \\ z \end{bmatrix} \tag{3.9}$$

また，式 (3.7) の v'-w'平面座標を v-w 座標に以下の計算式で変換する．

$$\begin{bmatrix} v \\ w \end{bmatrix} = \begin{bmatrix} \cos(roll) & -\sin(roll) \\ \sin(roll) & \cos(roll) \end{bmatrix} \times \begin{bmatrix} v' \\ w' \end{bmatrix} \tag{3.10}$$

最後に，式（3.9）および式（3.10）より，物体の三次元座標（x’, y’ z’）は以下の計算式で求められる．

$$\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = \begin{bmatrix} \sin(yaw) & \sin(pitch) \cdot \cos(yaw) \\ 0 & -\cos(pitch) \\ -\cos(yaw) & \sin(pitch) \cdot \sin(yaw) \end{bmatrix} \times \begin{bmatrix} v \\ w \end{bmatrix} + \begin{bmatrix} cx \\ cy \\ cz \end{bmatrix} \tag{3.11}$$

以上，式（3.8）と式（3.11）の結果から，物体名，物体の三次元座標，物体の実際の幅と高さ，カメラの位置姿勢およびカメラ情報をまとめてモデル配置部に送信する．

3.4.5 物体の VR 空間配置手法

物体位置計算部からデータを受け取ると，物体別で事前に用意された物体を VR 空間に配置していくが，この際，大きさが二次元であること，およびモデルの向きが取れないことが問題となる．ここで，物体の種類によって形が大体決まっているという条件を付ける．例えば，机や椅子は xz 方向の大きさがほとんど変わらない，モニタは幅があるが奥行きがほとんどない，理想的な角度から撮った場合に物体はカメラと 180 度向かい合わせになっている，などである．また，カメラはキャンパスを周回しているロボットに設置することを想定しているため，pitch は 0 度であると条件づけることができる．これらの条件より，以下の処理で配置するモデルの大きさおよび向きを決めた．

(1) 物体位置姿勢計算部からデータを受け取ると，カメラと 180 度向かい合わせの状況かつ，高さが y 軸の大きさとし，物体別に幅が xz 軸の大きさである，として VR 空間に配置

(2) 検出物体別，座標別，撮影時間別で三次元リストを作成

a. 撮影時間が異なっても，座標が等しいときは同じ物体と扱う

b. 一定以上近い物体は，座標ズレの範囲内として同じ物体と扱う

(3) 検出物体別に，座標別に正面と考えられる物体のみを描画

ここで，物体の正面判定は検出物体によって変わるが，例えば机や椅子は xz 軸の大きさが同じという条件より幅が最小のとき，正面を向いていると考えられ，モニタなどは奥行きがないため，幅が最大の時，正面を向いていると扱えると考えられる．例えば，キャンパスを動き回るロボットから時刻によって**図 3.3** に示すように，複数視点からの映像を撮ることができるが，この際，モニタは視点 2 のようにほぼ正面から撮った際に幅が最大化される．

以上の処理によって，動的環境を VR 空間に反映することができる．

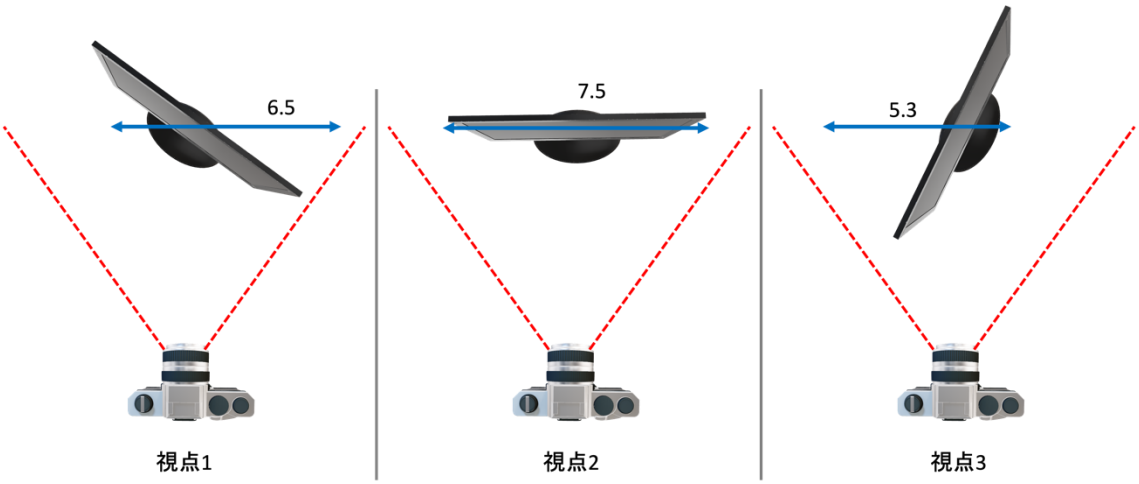


図 3.3　異なる視点から測定されるモニタの幅（比）

第4章 動作確認

本章では, 第 3 章で検討した通信および三次元位置計算手法の有用性を確認するために行なった動作確認について述べる.

4.1 目的

本論文で試作したシステムが構築する VR 空間について, MR キャンパスを実現するための実空間のデジタルツインとしてどの程度の性能を持っているかを評価する．具体的には以下に示す内容について確認する.

- ROS 通信における各トピックの通信量
- RGBD データを取得してから物体の配置までの，システム全体の処理時間
- 二次元物体検出アルゴリズムの精度
- 実空間の物体の実際の位置と，システムによって算出された位置の誤差
- 実空間の物体の実際の向きにより近い物体の配置ができているか

MR キャンパスはキャンパスという広範囲でシステムが動作するため, 複数台の周回ロボットおよび, 大多数のユーザが同時に大学の Wi-Fi を使用すると考えられる．したがって，システムの通信量は極力少ない方が良い. また, VR 空間が実空間のデジタルツインであるためには, データを取得した時刻にすぐ VR 空間に反映されることが理想である．しかし，今回は周回型のロボットで実装することを想定しているため，最大でも 1 日に数回ほど更新ができる処理時間であれば良い. そして, 二次元物体検出アルゴリズムはあらゆる角度から撮影しても正しく検出されることが理想である．加えて，3.1 節で述べたように，実空間の物体の位置と，VR 空間においてその物体を表すオブジェクトの位置および向きは一致していることが理想である.

4.2 セットアップ

今回使用したデバイスの仕様およびシステム構成を**表 4.1**，**表 4.2**，**図 4.1** に示す[15]．また，二次元物体検出アルゴリズムとして，今回は YOLOX を用い，学習済みデータセットモデルとしては YOLOX 公式の github で公開されている YOLOX-x を用いた[16]．ここで，YOLOX-x が認識する物体には，tv や dining table，chair，person，laptop，mouse など計 80 種あり，今回は tv についてのみ実装をした．将来的には必要なモデルを重点的に学習したデータセットを使うことで様々な物体，状況に対応することができると考えられる．RGBD カメラとして使用した RealSense D435 は USB 3.0 で OS として Ubuntu 22.04 を搭載した PC に有線接続されており，各 PC は Ethernet ケーブルで LAN に有線接続されている.

RealSense D435 のセンサデータの送信方法としては RealSense 公式の realsense-ros という，RGB 深度センサの視野角を RGB センサの視野角にキャリブレーションした上で ROS トピックとして任意の解像度，fps で送信できるものを用いた[17]．そして，YOLOX も同様に YOLOX 公式の YOLOX-ROS を利用した[18]．ここで，realsense-ros が送信する画像のトピッ

表 4.1　使用した PC の性能

OS	Ubuntu 22.04 LTS	Ubuntu 20.04 LTS	Windows 10 Pro
CPU	Intel Core i7-13700F CPU @5.20GHz	Intel Core i7-7700K CPU @4.20GHz	Intel Core i5-6400 CPU @2.70GHz
GPU	NVIDIA GeForce 4070	NVIDIA GeForce RTX 1080	NVIDIA GeForce RTX 3060
RAM	32GB	16GB	24GB

表 4.2　使用した RGBD カメラの仕様

商品名		RealSense Depth Camera D435
型番		82635AWGDVKPRQ
RGB センサ	最大解像度	1920 x 1080 (30fps)
	視野角	水平 69 度，垂直 42 度
Depth センサ	最大解像度	1280 x 720 (90fps)
	視野角	水平 87 度，垂直 58 度
	最小深度	28cm
	最大深度	10m

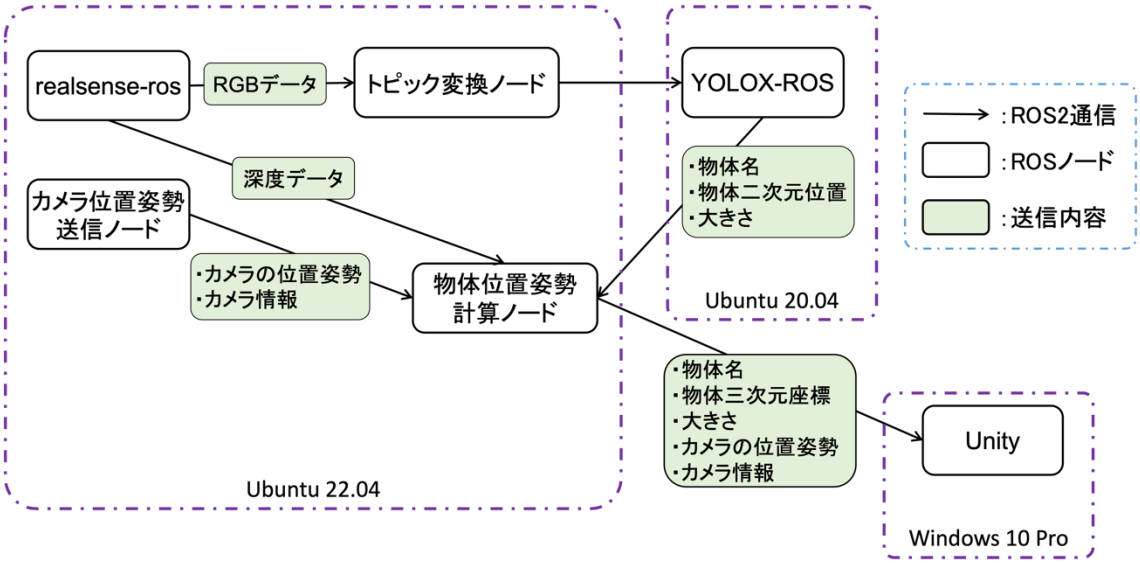


図 4.1　動作確認時のシステム構成

クと YOLOX-ROS が受け取る画像のトピックが異なることから，間にトピックを変換するノードを挟んでいる．また，RGBD データはどちらも解像度 640 x 480 (15 fps) に設定した．加えて，RealSense D435 は将来的に周回型ロボットに乗せることを想定しているため，自己位置推定ができると考えられるため，今回は手動でカメラの位置姿勢を送信するノードを自作した．他に，Unity と ROS2 間の通信には 3.2 節で紹介した Unity-Robotics-Hub を用いた．

4.3　通信に関する動作確認の結果

本節では，図 4.1 を参考に各トピックの通信量を測定したものと，システム全体の処理時間について記述する．

4.3.1　各トピックの通信量

ROS2 のコマンド”ros2 topic bw”を用いて，約一分間測定を行った各トピックの平均通信量について**表 4.3** に示す．ここで，表のトピック名 a ~ f は図 4.1 におけるノード間のトピックを表し，それぞれどのノード間のトピックを指すかを以下に示す．

a. realsense-ros が配信している RGB データのトピック
b. realsense-ros が配信している深度データのトピック
c. トピック変換ノードが配信している RGB データのトピック
d. YOLOX-ROS が配信しているデータのトピック
e. カメラ位置姿勢送信ノードが配信しているデータのトピック
f. 物体位置姿勢計算ノードが Unity に送信しているトピック

4.3.2　システム全体の処理時間

RGB データ送信から，Unity 内の VR 空間内にモデルを配置する処理が完了するまでのシステム全体の処理時間を複数回計測した結果を示す（**図 4.2**）．

4.4　二次元物体検出アルゴリズムの動作確認の結果

今回使用した二次元物体検出アルゴリズムである YOLOX および，データセット YOLOX_x について動作確認を行った結果を示す（**図 4.3**）．

表 4.3　各トピックの通信量平均

トピック名	a	b	c	d	e	f
通信量平均	5.62 MB/s	2.26 MB/s	49.63 KB/s	348 B/s	76 B/s	441 B/s

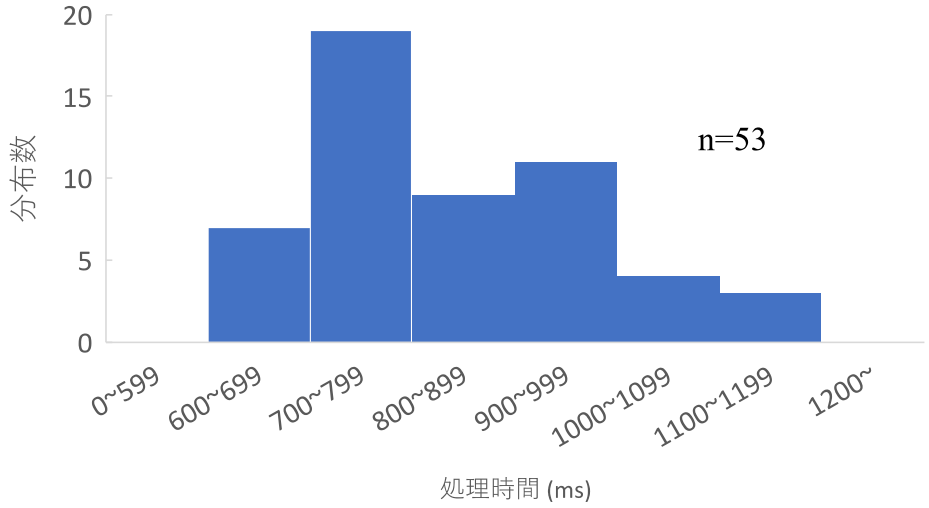
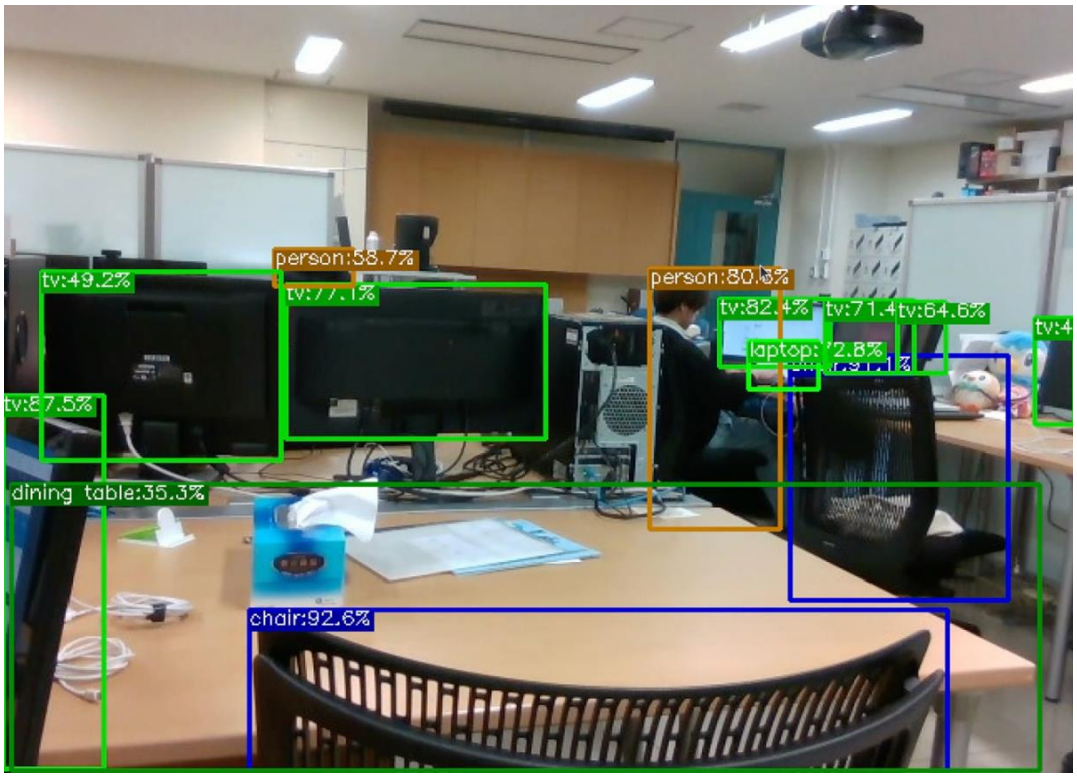
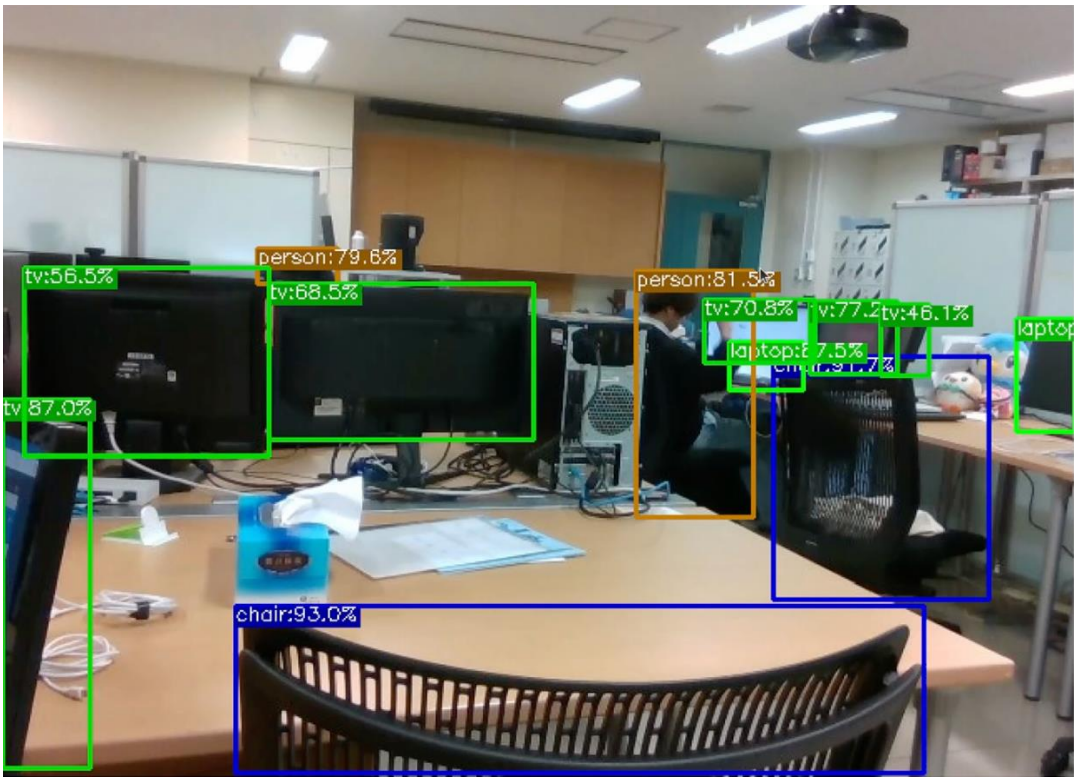


図 4.2　システム全体の処理時間分布



(a) 机の認識ができている場合



(b) 机の認識ができていない場合

図 4.3 YOLOX 動作確認結果

4.5　物体の三次元位置姿勢および大きさ計算手法に関する動作確認の結果

本節では，物体の三次元位置について実空間との誤差を測り，複数視点から撮影した画像データから正しい向きを判断できているか，および深度や物体の大きさの誤差について確認する．ここで，今回の動作確認環境および，測定するカメラやモニタの実空間における位置，向き，大きさを示す（**図 4.4**，**表 4.4**）．今回，実空間の座標系として，図 4.4 における部屋の一番奥にあたる部分を原点として，右側を x 軸，高さを y 軸，奥行きを z 軸とし，3 パターンのカメラの位置姿勢から手前の 3 つのモニタの位置姿勢および大きさについて測定する．

4.5.1　物体までの深度

表 4.4 から求められるモニタとカメラの距離と，RealSense から取得した深度情報を**表 4.5** に示す．

4.5.2　算出された物体の三次元位置姿勢および大きさ

今回提案した手法によって算出された各カメラ位置からのモニタの位置および大きさの実空間との誤差を示す（**表 4.6**）．

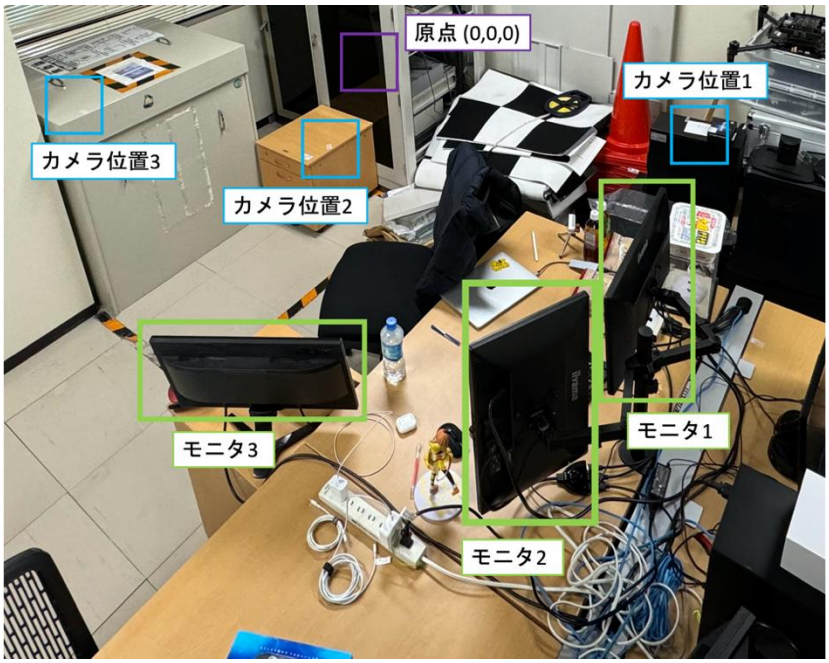


図 4.4　動作確認環境

表 4.4　モニタ及びカメラの位置姿勢

	x (m)	y (m)	z (m)	yaw (°)	roll (°)	pitch (°)	幅 (m)	高さ (m)
モニタ 1	2.0	1.2	3.2	270	0	0	0.5	0.3
モニタ 2	2.5	1.2	3.2	225	0	0	0.3	0.5
モニタ 3	2.7	1.0	2.5	225	0	0	0.5	0.3
カメラ 1	0.7	1.0	3.0	0	0	0		
カメラ 2	1.4	0.7	1.3	45	0	0		
カメラ 3	2.3	1.3	1.0	90	0	0		

表 4.5　カメラとモニタ間の距離

(a) 実際の距離（m）

	モニタ 1	モニタ 2	モニタ 3
カメラ 1	1.33	1.82	2.06
カメラ 2	2.05	2.25	1.79
カメラ 3	2.22	2.21	1.58

(b) RealSense から取得した距離（m）

	モニタ 1	モニタ 2	モニタ 3
カメラ 1	1.123	1.867	1.982
カメラ 2	2.055	2.315	1.812
カメラ 3	2.485	2.491	1.568

4.6　考察

本節では 4.3 節から 4.5 節の動作確認結果を分析し，今後の課題などについて考察する．

4.6.1　通信量に関する考察

トピック c ~ f の通信量に関しては MR キャンパスの実現にあたって十分に少ないといえる．理由としては，c の場合 YOLOX-ROS が受け取るトピックが圧縮された RGB データであるため，d ~ f は画像のような大容量データの通信を行っていないためと考えられる．一方，トピック a, b は RGBD データの送信トピックであるためキャンパスの周回ロボットの数が数十台規模になると，合計で数 100MB/s 以上の通信を行うことになり，安定して Wi-Fi で通信を行うことは困難であると考えられる．解決策としては，realsense-ros ではなく，始めから RGB データと深度データをキャリブレーションした上で，圧縮して送信するようなノードを自作することである．これによって，トピック c と同程度の通信量まで削減でき，MR キャンパスの実現にあたって問題がなくなると考えられる．

4.6.2　システム全体の処理時間に関する考察

前提として周回型のロボットの周回頻度は現実的に考えて多くとも 1 日数回程度であるため，処理時間としては数分単位で行うことができれば十分である．ここで，図 4.2 より全て数秒単位で処理が完了しているため，処理時間に関しては問題ないと考えられる．

4.6.3　二次元物体検出アルゴリズムに関する考察

図 4.3 より，今回用いたデータセットではモニタや椅子は検出される確率が高いが，机の検出率は低かった．しかし，MR キャンパスを実装する場所に特化したデータセットを作成すれば解決すると考えられる．

4.6.4　物体の三次元座標計算手法に関する考察

各カメラからモニタまでの深度について表 4.5 より，おおよそ誤差 0.2m 程度と判断でき，三次元位置計算するにあたって少なくとも 0.2m 程度の誤差が生まれる．これは，二次元画像内の物体の位置を判定する際に，YOLOX が返すバウンディングボックスで囲まれた部分の中心を参考にしているため，向きによっては正しい位置が取れていないといったことが原因であると考えられる．また，三次元位置に関しては表 4.4 および表 4.6 より，基本的に x, y, z 全ての値に関しておおよそ最大 0.2m 程度の誤差が生じている．これは，深度情報の誤差に加えて，先ほど同様に物体の二次元座標の取得時点で誤差があると考えられる．最後に，大きさに関しては表 4.4 および表 4.6 より，高さに関しては誤差 0.1m 程度でており，幅に関しては確かに向きによって

表 4.6　各カメラから算出された位置および大きさの誤差

(a) カメラ 1

	x (m)	y (m)	z (m)	幅 (m)	高さ (m)
モニタ 1	-0.10	-0.07	-0.06	-0.41	0.10
モニタ 2	0.05	-0.14	-0.09	-0.15	0.13
モニタ 3	0.06	-0.02	0.10	-0.20	0.06

(b) カメラ 2

	x (m)	y (m)	z (m)	幅 (m)	高さ (m)
モニタ 1	-0.13	0.00	-0.22	-0.16	0.13
モニタ 2	-0.18	-0.02	-0.08	-0.08	0.07
モニタ 3	0.02	0.08	-0.04	-0.12	0.05

(c) カメラ 3

	x (m)	y (m)	z (m)	幅 (m)	高さ (m)
モニタ 1	0.11	-0.02	0.18	-0.15	0.07
モニタ 2	-0.04	-0.04	0.20	-0.09	0.07
モニタ 3	-0.11	0.04	0.14	-0.22	0.14

異なる値が取得できており一番大きな値がより妥当な向きから取れていることがわかる．しかし，三次元座標の誤差に合わせて，異なる視点から取られた物体を同じ物体とみなすための座標の誤差閾値を設定する必要があり，この閾値の距離より近い物体同士を別の物体であると認識することが困難となる．また，カメラ 2 とカメラ 3 から取得されたモニタ 2 およびモニタ 3 の幅はわずかに妥当な向きの方が大きいが，ほぼ同じ値が算出されている．

したがって，本論文で提案したシステムを導入しても，デジタルツインの構築に関しては依然として課題があり，特に物体の向き検出に関しては改善していく必要があると考えられる．具体的には，位置検出の精度を上げて閾値を下げる他に，計算された幅から物体の向きを算出するのではなく，物体の向き自体を直接算出する必要があると考えられる．

第5章 むすび

本論文では, 現実世界をより豊かにするリアルワールドメタバースをコンセプトにした，MR キャンパスという構想を紹介し，その中で必要となる技術である実空間という動的環境のデジタルツインを構築する手法の提案を行なった．具体的には，実空間のデジタルツインとしての VR 空間を構築する際，よりリアルタイム性が高いことが求められるが，モデリングや三次元再構成などではこれが困難であるという問題に対して，事前にデジタルツインを実現する上で必要な物体を全てモデリングした上で，一定頻度でキャンパスをセンシングし物体及びその位置姿勢を検出することで，VR 空間に代替的な 3D モデルを VR 空間に配置するという手法を考案した．この手法を用いることで，1 日に最大数回程度ではあるものの前に述べた二つの手法に比べて高頻度で容易に動的環境を反映することができると考えられる．

提案された手法にしたがって，動作確認環境を構築した結果，RGBD データの送信に関しては改善の余地があるものの十分少ない通信量で通信できていることや，物体の三次元座標を誤差 0.2m 程度で算出できることを確認した．

しかし，物体の向きの反映に関しては，三次元座標の誤差がクリティカルとなり，同じ種類の物体が近くに存在した場合, 本提案手法では異なる物体を同物体であると判定してしまうといった問題が発生した．

今後の展望としては，座標を参照して向きを設定するのではなく，向きを直接取得する手法について検討し，その上で三次元位置の計算手法に関して，二次元物体検出アルゴリズムでの画像における位置算出の際に，物体の向きから射影変換を用いてバウンディングボックスにおける物体の中心を求めるように改良することが挙げられる．

謝　辞

本研究を通して，以下の方々にご指導とご協力をいただきました．厚くお礼申し上げます．

立命館大学　情報理工学部　情報理工学科　柴田史久　教授

立命館大学　情報理工学部　情報理工学科　木村朝子　教授

立命館大学　情報理工学部　情報理工学科　中村文彦　助教

立命館大学　立命館グローバル・イノベーション研究機構　橋口哲志　准教授

同研究グループの梶山春香氏，岡田大翔氏，堺亮太氏，LIU Jinyou 氏，江崎佑真氏，大内健太郎氏，矢野永真氏，大橋正明氏，YUN Jonghee 氏，小林良偉氏，長尾拓実氏，吉野智氏

モバイルコンピューティング研究室の皆様，リアリティメディア研究室の皆様

特に，梶山春香氏，江崎佑真氏には同研究グループの先輩としてお世話になりました．心より深く感謝申し上げます．

参考文献

[1] 舘暲, 佐藤誠, 廣瀬通孝 : “バーチャルリアリティ学”, 日本バーチャルリアリティ学会, 2011 年.

[2] バーチャル美少女ねむ : “メタバース進化論”, 技術評論社, 2022 年.

[3] 総務省情報通信政策研究所調査研究部 : “Web3 時代に向けたメタバース等の利活用に関する研究会 報告書”, https://www.soumu.go.jp/main_content/000892205.pdf, 2023 年.

[4] 吉田塁 : “東京大学「メタバース工学部」における取組み”, 電子情報通信学会 通信ソサイエティマガジン, Vol. 17, No. 3, pp. 230 - 234, 2023 年.

[5] 矢野経済研究所, “メタバースの国内市場動向調査を実施（2023 年）”, https://www.yano.co.jp/press-release/show/press_id/3333 （2024 年 1 月 4 日）

[6] Y. Ohta and H. Tamura (eds.): “Mixed Reality Merging Real, and Virtual Worlds,” Ohm sha & Springer Verlag, 1999.

[7] B. MacIntyre and M. A. Livingston (eds.): “(Special Session) Moving Mixed Reality into the Real Worlds,” IEEE Computer Graphics, and Applications, Vol. 25, No. 6, pp. 22 - 56, 2005.

[8] Faisal Zaman, Craig Anslow, Andrew Chalmers and Taehyun Rhee: “MRMAC: Mixed Reality Multi-user Asymmetric Collaboration” IEEE, 2023.

[9] Microsoft: “Dynamics 365 Remote Assist,” https://dynamics.microsoft.com/ja-jp/mixed-reality/remote-assist/ （2024 年 1 月 1 日）

[10] PTC: “Vuforia Chalk,” https://www.ptc.com/ja/products/vuforia/vuforia-chalk （2024 年 1 月 1 日）

[11] David Lindlbauer, and Andrew D. Wilson: Remixed Reality: “Manipulating Space and Time in Augmented Reality,” C*HI,* No. 129, pp. 1 - 13, 2018.

[12] Benran Hu, Junkai Huang, Yichen Liu, Yu-Wing Tai, and Chi-Keung Tang: “NeRF-RPN: A general framework for object detection in NeRFs,” *Computer Vision and Pattern Recognition*, pp. 23528 - 23538, 2023.

[13] 藤重秀斗, 松室美紀, 木村朝子, 柴田史久 : “歩行者飛び出し予測のための周辺車両の半隠消表示法”, TVRSJ, Vol. 28, No. 3, pp. 141 - 151, 2023 年.

[14] Unity-Technologies/Unity-Robotics-Hub : https://github.com/Unity-Technologies/Unity-Robotics-Hub/tree/main （2024 年 1 月 31 日）

[15] Intel Corporation: Intel® RealSense™ Depth Camera D435, https://www.intelrealsense.com/depth-camera-d435/ （2024 年 1 月 7 日）

[16] Megvii-BaseDetection/YOLOX : https://github.com/Megvii-BaseDetection/YOLOX （2024 年 1 月 26 日）

[17] IntelRealSense/realsense-ros :

https://github.com/IntelRealSense/realsense-ros （2024 年 1 月 26 日）
[18] Ar-Ray-code/YOLOX-ROS：
https://github.com/Ar-Ray-code/YOLOX-ROS （2024 年 1 月 26 日）