Followingraphic Display: 実物体に追従する空中像

菅原 陵央[1]　林 竜吾[1]　小泉 直也[1,a)]

概要：本稿では，空中像を実物体に追従させるインタラクション手法を提案する．光源をアクチュエータで動かすことで，空中像を奥行き方向に移動させることが可能である．しかし，空中像が目標位置に到達したというフィードバックを待つ設計にすると，常に座標が変化する実物体に同期させて動かすことが難しい．本研究では，アクチュエータの制御において待ちを入れず，速度制御を高速化することで，空中像を実物体の動きに追従させるシステムを提案する．評価として，Motion-to-photon Latency を測定した．その結果，ディスプレイ表示及びアクチュエータ駆動の遅延は平均 23 ms 程度であることが確認された．

1. はじめに

空中像は，光源から出た光線を再び別の一点に集めることで形成される実像である．プロジェクションマッピングのような表示面を必要とせず，単独で空間中に浮かんだ映像を表示できる．そのため，空中像は実物体の隣に表示可能であり，実空間とデジタル映像をシームレスに融合できる．

空中像と実物体を一体として動かすようなインタラクションを考えた場合，空中像を実物体の動きに対して追従させる必要がある．例えば，空中像キャラクターをブランコに隣接表示した状態で動かしたいと考えたとき，キャラクターがブランコに追従しなければならない．そのためには，実物体の位置をセンシングし，その結果を空中像の動きに反映させる必要がある．

従来の研究では，空中像を奥行き方向へ移動させるために，リニアアクチュエータを用いる手法や [1]，光源と一体となった実物体を動かす手法 [2] などが提案されている．前者の場合は，固定された実物体に空中像が遅れて向かう形式のインタラクションのみ実現されており，動く実物体に追従するものは提案されていない．後者の場合は，空中像が鉛直方向に移動できないという課題がある．

本研究では，奥行き方向の移動にアクチュエータを用いた際，速度の更新を高速に行うことで，空中像を実物体の動きに追従させるシステムを提案する．アクチュエータからのフィードバックを待たずに，トラッキングごとに速度を更新することで，実物体の動きに同期するような空中像インタラクションを実現する．

[1] 電気通信大学
[a)] koizumi.naoya@uec.ac.jp

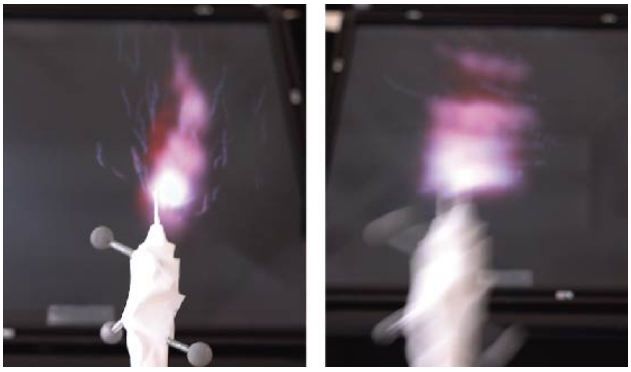

図 1　ろうそくの火として表示した空中像を空間移動させた様子

2. 関連研究

2.1 空中像光学系

空中像を形成する光学系として，Micro Mirror Array Plate（MMAP）[3] を利用した光学系がある．MMAP は等間隔で並んだミラーが直交するように 2 層で重なった構造をしており，光源に対して面対称な位置に実像を形成する．そのため，光源を動かすことで，空中像を動かすことができる．

本研究では，MMAP を用いた光学系を採用した．地面に対して水平にディスプレイを置き，MMAP を 45 度傾けて配置すれば，ユーザーから見て正面に空中像が形成される．そのため，ディスプレイ内の表示位置を変えることで，空中像を水平・鉛直方向に移動でき，ディスプレイをアクチュエータで上下させることで，空中像を奥行き方向に移動できる．

2.2 空中像の空間移動を利用したインタラクション

高崎らは，手を用いた空中像インタラクション手法を提

案している [4]．手によって空中像が遮蔽されるオクルージョン問題を，手へのプロジェクションで解決している．しかし，光源を奥行き方向に動かしていないため，空中像の奥行き方向への移動範囲に制限がある．

Kim らは，空中像キャラクターが空間内を移動するシステム MARIO を開発した [1]．水平・垂直方向の動きに加えて，リニアアクチュエータを用いて奥行き方向の動きを実現している．しかし，空中像が実物体の最高点に向かってジャンプする形式で移動するよう設計されており，実物体の動きに同期させて動かすことはできない．

梶田らは，運動する実物体に空中像を隣接表示する際の遅延による位置ずれを解決するために，センシングを用いない手法を提案した [2]．MMAP とミラーを用いた光学系により，遅延なく空中像を動かすことに成功した．しかし，鉛直方向の動きに対応していない．また，実物体と光源が一体となっているため，実現できるインタラクションの種類が限定的である．

本研究では，空中像を 3 次元に動かすことができる MARIO の光学系を用い，その上で，アクチュエータの制御を高速化することに焦点を当てる．実物体の位置をトラッキングし，それに合わせてアクチュエータを動かすことで，3 次元移動する実物体に対して，空中像を追従させることができる．

3. システム

3.1 リニアアクチュエータの制御

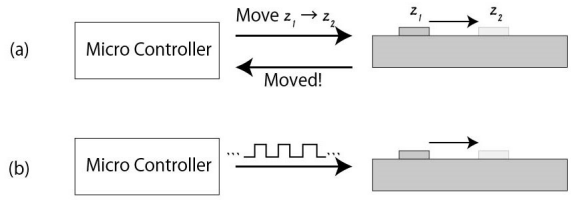

図 2 アクチュエータ制御方法の違い ((a) は目標位置を指定し，移動完了のフィードバックを待つ形式，(b) はフィードバックを待たずにパルス波を送り続ける形式)

本研究では，図 2 に示すように，アクチュエータの移動完了のフィードバックを待たずに速度を更新することで，制御の高速化を図る．アクチュエータの動きを実物体の動きに同期させるには，トラッキングごとにその都度アクチュエータの目標位置を更新する必要がある．最初に目標位置と速度を定め，移動完了したことが分かった後にいったん運転を止め，再び位置・速度を指定して駆動する制御方法では，実物体の動きに同期させることが難しい．

リニアアクチュエータは，パルス波によって駆動される．1 パルスにつき 1 ステップ分進むため，パルス周期を短くするほど速度が上がる．位置は，生成したパルス数を数えることで分かる．

3.2 実装

システム図を図 3，実装した筐体を図 4 に示す．最初に，光学式モーションキャプチャで実物体がトラッキングされる．そこで取得された座標はゲームエンジンに送信される．ゲームエンジン内部では，2 つのスレッドに分かれて処理を行う．1 つ目のスレッドでは，空中像の水平・垂直方向の移動に関連した処理が行われる．このスレッドを含む経路をディスプレイ系統と呼ぶことにする．2 つ目のスレッドでは，空中像の奥行き方向の移動に関連した処理が行われる．このスレッドを含む経路をアクチュエータ系統と呼ぶことにする．

光学式モーションキャプチャとして OptiTrack V120: Trio，ゲームエンジンとして Unity を用いた．OptiTrack は 120 fps の赤外線カメラで実物体をトラッキングし，専用のソフトウェア Motive 内で位置や姿勢が計算される．その後，データ化された座標がストリーミングされ，Unity に送られる．Unity では，targetFrameRate を 360 fps に，maxQueuedFrames を 2 に設定した．

ディスプレイ系統では，空中像をディスプレイに表示するための処理が行われる．Thunderbolt4 対応の USB ケーブルで接続されたディスプレイに，レンダリング結果が表示される．なお，ディスプレイは JAPANNEXT 製の JN-MD-173GT240FHDR（240 Hz，17.3 インチ，応答速度は最大 4 ms）を使用した．

アクチュエータ系統では，Unity で計算されたパルス幅がシリアル通信でマイクロコントローラーに送られる．式 (1) に，パルス幅の計算式を示す．

$$w = \frac{a\Delta t}{\Delta z} \tag{1}$$

w はパルス幅，Δt はトラッキングの時間間隔，Δz は実物体の奥行き方向の座標変化，a はトラッキング座標をステップ数に変換する係数である．また，実物体のトラッキングが行われていない間は，アクチュエータの加減速処理が行われる．加減速処理では 1 ms ずつ，トラッキング結果に基づいて計算されたパルス幅に到達するよう調整される．

マイクロコントローラーで生成されたパルス波がモータードライバーに入力され，アクチュエータの駆動電流が生じる．マイクロコントローラーは Raspberry Pi Pico を用いた．モータードライバーはオリエンタルモーター社製の AZD-KX，リニアアクチュエータは同社製の EASM4RNXE040AZAK（ストローク 400 mm，最高速度 300 mm/s）を用いた．

4. 評価

システムの遅延を評価する指標として Motion-to-photon

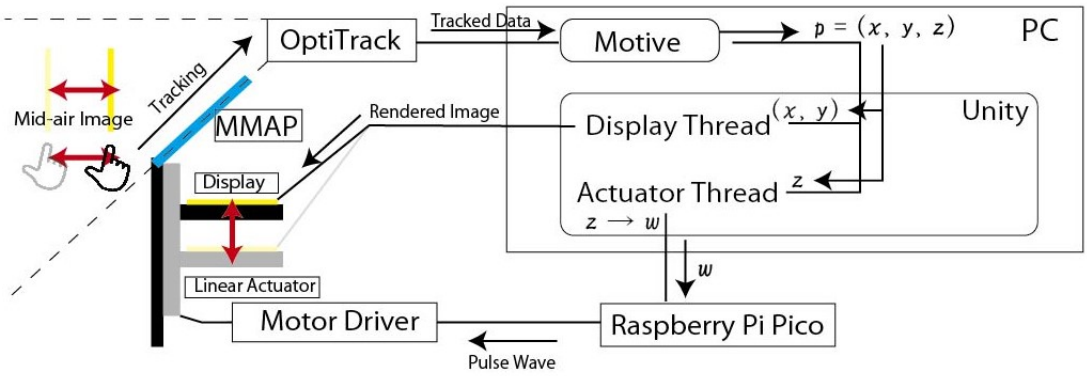

図 3 システム図

図 4 実装した筐体

図 5 空中像が動き出したときに白から赤に色を変える

Latency を用いた．Motion-to-photon Latency は，物体の運動がトラッキングされてから視覚フィードバックに反映されるまでの時間差であり [5]，HMD の遅延評価などで用いられている [6]．本研究では，ディスプレイ系統，アクチュエータ系統の Motion-to-photon Latency をそれぞれ測定した．

4.1 実験方法

実物体として，上面にマーカーを 3 つ付けた箱を用いた．箱を動かす際，指で弾いて初速を与えた．なお，OptiTrack のトラッキング結果にはノイズが含まれており，実物体が止まっている状態でも 0.03 mm ほど変動する．そのため，座標変化が 0.04 mm を超えたとき，空中像を動かすようにした．

ディスプレイ系統の遅延測定では，実物体と空中像が同時に撮影できる位置にカメラを設置した．カメラは Sony 製の RX10II(DSC-RX10M2) を使用し，960 fps で撮影した．なお，図 5 に示すようにして，空中像の動き始めに色を白から赤に変える処理を入れた．つまり，実物体が動き始めてから空中像の色が赤に変わるまでの時間を，ディスプレイ系統における遅延時間とした．測定回数は 20 回であった．

撮影した映像に対して，動画編集ソフトでフレーム数をカウントした．動画編集ソフトは Adobe Premiere Pro 2024 を使用した．実物体の動き始めから空中像の動き始めまでのフレーム数をカウントし，1 フレーム分の時間 $1000/960 \simeq 1.042$ ms を掛けたものを遅延時間とした．

アクチュエータ系統の遅延測定では，動き始めと動き終わりの両方を測定した．実物体とアクチュエータが同時に撮影できる位置から撮影し，同様にフレーム数をカウントした．なお，960 fps では，アクチュエータの動き始め・動き終わりを目視で判別することが難しかったため，動画編集ソフトで 240 fps に変換したうえで，フレーム数を数えた．1 フレーム分の時間 $1000/240 \simeq 4.167$ ms をかけたものをアクチュエータ系統の遅延時間とした．測定回数は 20 回であった．

4.2 結果

Motion-to-photon Latency の測定結果を箱ひげ図を用いて図 6 に示す．ディスプレイ系統の平均は 22.4 ms，中央値は 22.9 ms，最大値は 28.1 ms，最小値は 16.7 ms であった．アクチュエータ系統（動き始め）の遅延時間の平均は

23.1 ms，中央値は 22.9 ms，最大値は 29.2 ms，最小値は 16.7 ms であった．アクチュエータ系統（動き終わり）の遅延時間の平均は 22.9 ms，中央値は 25 ms，最大値は 29.2 ms，最小値は 16.7 ms であった．ディスプレイ系統，アクチュエータ系統ともに同程度の遅延が生じていることが分かった．また，アクチュエータ系統では，動き始めと動き終わりの遅延も同程度であることが確認された．

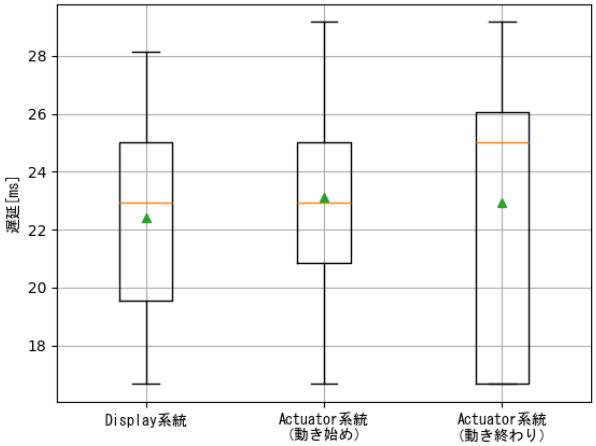

図 6 Motion-to-photon Latency の測定結果

5. アプリケーション

図 7 奥行き方向に動くブランコに乗った空中像キャラクター（上図はブランコが手前に移動した瞬間，下図はブランコが奥に移動した瞬間）

図 7 にアプリケーションの制作例を示す．揺れるブランコに空中像キャラクターを乗せて動かしている．本研究において実装したシステムを用いることで，映像を実物と同じようにして実物体に乗せることができるアプリケーションや，図 1 のように，ユーザーが主体的に空中像を操作できるインタラクションが実現できる．

6. まとめ

本研究では，動く実物体に空中像を追従させるインタラクションを実現した．アクチュエータの移動完了のフィードバックを待たずに速度を更新することで，実物体の動きに追従させることに成功した．実装したシステムにおける Motion-to-photon Latency を測定した結果，ディスプレイ表示及びアクチュエータ駆動で平均 23 ms 程度の遅延があった．今後の課題として，センシングをマーカーレスにする試みや，実物体との位置関係によるオクルージョン問題の解決が挙げられる．

謝辞 本研究は，JST 創発的研究支援事業 JPMJFR216L の支援を受けたものです．

参考文献

[1] Hanyuool Kim, et al. Mario: Mid-air augmented reality interaction with objects. *Entertainment Computing*, Vol. 5, No. 4, pp. 233–241, 2014.
[2] Hajime Kajita, et al. Skyanchor: Optical design for anchoring mid-air images onto physical objects. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology*, pp. 415–423, 2016.
[3] 山本裕紹（編）. 操作・検査のタッチレス化/非接触化のための設計ポイントと最新動向. 情報機構, 2020.
[4] 高崎真由美ほか. Mmap を用いた空中映像と実物体との位置関係を明瞭にする表現手法の検討. マルチメディア，分散協調とモバイルシンポジウム 2021 論文集, Vol. 2021, No. 1, pp. 1114–1119, 2021.
[5] Matthew Warburton, et al. Measuring motion-to-photon latency for sensorimotor experiments with virtual reality systems. *Behavior Research Methods*, Vol. 55, No. 7, pp. 3658–3678, 2023.
[6] Chun-Ming Chang, et al. Performance measurements of virtual reality systems: Quantifying the timing and positioning accuracy. In *Proceedings of the 24th ACM International Conference on Multimedia*, MM '16, p. 655–659, New York, NY, USA, 2016. Association for Computing Machinery.