ダイナミック・マニピュレーションにおける状態予測と動作生成に関する研究

松 嶋 道 也

大阪大学大学院 基礎工学研究科

2005年1月

論文要旨

当初，工場内における部品の把持や，ハンドリングなど，与えられた動作をただ正確にこなすだけであった産業ロボットは，様々な知覚センサーを用いて外界の環境の情報を取得することにより，柔軟かつ正確な作業が行えるようになっており，特に最近のロボット工学の分野では，工場の生産ラインのみならず，人と同じ環境で活動できるようなロボットが現れている．とくに，近年のコンピュータの著しい計算速度の向上や，記憶装置の価格低下により，数年前までは高価な専用装置を必要とした画像処理や画像解析がより身近な物になってきた．ところが，人間の生活環境において，変化というものは避けられない．ダイナミックに変化する環境において，ロボットが環境に適応してタスクを行う場合，ロボットは得られる環境の情報を処理し，環境に適した動作を，連続的に動作を行いながらタイミングを合わせて実行しなければならない．動的な環境への適応能力は人間環境のみでなく，工学的／産業的な面から見ても重要である．こうした背景のもと，人が外界の刺激と運動を関連づけることで実現するダイナミックマニピュレーションタスクとして，卓球を取り上げて研究を行った．本研究では人の動作とターゲットの変化に対する対応の手段をロボットに適用することにより，ダイナミックに変化するターゲットへの適応能力を開発することを目的とする．スポーツ科学の分野では，ボール打撃時のラケットスピードや位置を仮想ターゲットと呼び，ストローク動作はこの仮想ターゲットの予測と微調整に基づいて実行されるという仮説が提案されている．そこでまず，このアイデアをロボットによって具体化する手法について述べる．仮想ターゲットの予測には k dimensional tree(k-d tree)と呼ばれるデータ構造を利用した入出力マップを，仮想ターゲットに基づく主要なストローク動作の生成には Koditschek のミラー則（mirror law）に類似の視覚フィードバック制御を用いて卓球タスクを行った．しかしながら，視覚フィードバックに基づく制御で，急激な動作を行うとサーボ誤差やシステムの弾性の問題から，実際の動作は目標の軌道に対して遅れを生じる．このため，動作が不安定になることがある．人間は卓球のような動的なタスクを行う際に，人間は変化するターゲットへの対応を熟練によってこなし，飛来するボールを適切に打ち返すために必要な動作を身につける．更にその行動を繰り返すことで無駄なくスムーズに作業をこなすことができるようになる．そこで次に，このような人間のスキルに倣い，繰返しタスクを行う中でデータを蓄積し，より適切な打撃動作を計画，実現できる学習手法を提案する．この学習は，飛来するボールに対する適切な打撃動作を計画するためのものと，計画された打撃動作を正確に実現するためのものの 2 つがある．前者はボールの飛行時や打撃時のボールの挙動が打撃動作に依存し，かつ再現性があることを利用して，LWR を用いたマップにより実現する．後者は，システムの線形性を利用し，新たな軌道を正確に実現する入力を繰返し学習を行わないで求める学習制御の応用手法を用いて実現する．最後に，実際の人間の動作を計測し，その再現性やタイミングに注目して，ロボットが人間の動作から，より直接的にダイナミックマニピュレーションを学習する手法を提案する．一定動作部分をあらかじめマスタースレーブシステムを用いて抽出しておき，そのパターン動作を行うタイミングや開始位置を，再びマスタースレーブにより教示する．この結果を LWR マップに反映してボールに合わせた動作計画を行い，前半と同様に，新しい学習制御を用いてそれを実現する．

目 次

図 目 次

表 目 次

第 1章　序論

1.1　研究背景

当初，工場内における部品の把持や，ハンドリングなど，与えられた動作をただ正確にこなすだけであった産業ロボットは，様々な知覚センサーを用いて外界の環境の情報を取得することにより，柔軟かつ正確な作業が行えるようになっており，特に最近のロボット工学の分野では，工場の生産ラインのみならず，人と同じ環境で活動できるようなロボットが現れている [1, 2].

しかし，これらのロボットは外界に働きかけをするのではなく，自分自身のみで完結する動作のみを行うか，静的な環境もしくは位置を完全に限定された物体に対してのみなんらかの働きかけをすることができる.

ダイナミックに変化する環境において，ロボットが環境に適応してタスクを行う場合には，ロボットは得られる環境の情報を処理し，環境に適した動作を連続的に動作を行いつつ，タイミングを合わせて行わなければならない．ロボットをより環境に適応させるためには，視覚や触覚などの膨大な情報を手際よく処理し，記憶してサーボループに結合させる基本ルートを見つけること，感覚から運動への明確な手がかりを見付けることが重要である [3]．人は環境から得られた刺激情報を処理し，運動という形で再び環境に影響を与える．まず，刺激の特徴を予備的な知識を用いて解析し，刺激の列からパターンを抽出する．次に，刺激と運動の連結を行い，刺激情報に対する行動を選択する．そして最後に，実際の行動を行うための筋肉の運動へと変換する [4]．我々は，このような刺激に対する対応能力を必要とするタスクとして卓球を取り上げて研究を行ってきた.

人間は変化するターゲットへの対応を熟練によってこなし，更にその行動を繰り返すことで無駄なくスムーズに作業をこなすことができるようになる．この点に注目し，人の動作とターゲットの変化に対する対応の手段をロボットに適用することにより，ダイナミックに変化するターゲットへの適応能力の開発を目的とする.

Andersson は人間と卓球する高度なロボットシステムを構築した [5]．Andersson は，ボールやロボットの動特性の陽なモデルを利用してボールの打撃動作生成の問題をボールの状態予測とラケットの軌道生成の問題に分離し，ボールの状態の逐次予測とそれに基づくラケットの目標軌道の更新によってハイブリッド制御問題に対処する方法を示した．しかし，ロボットが生成できるラケット速度や加速度に限界があるため，実際にはヒューリスティックに設定した例外処理を多用している．Andersson の方法は，タスクや環境の陽なモデルとして人間の知識を大いに利用するものであり，そのパフォーマンスはタスクや環境，およびシステムの開発者の知識に依存する．言い換えれば，ロボットシステムは練習と経験を通じて，自分のスキルを向上させることはできない.

スポーツ科学の分野においては，Ramanantsoa は，経験者が卓球の打撃動作を計画して

実行する際に動作の自由度を制限するという Bernstein の仮定に基づいて，卓球の手順を単純化することを提案した [6]．１ストロークの動作を４つのフェーズに分割し，各フェーズごとにボールの動きに合わせて時空間的な動作調整を行うモデルを提案している．この案の本質は，ボール打撃時のラケット速度や位置を仮想ターゲットと呼び，ストローク動作はこの仮想ターゲットの予測と微調整に基づいて実現されるということである．本論文では，このような考え方をロボットによって具体化し，仮想ターゲットの予測と実現によってダイナミックマニピュレーションを実現する手法について述べる．

1.2　ミラー理論を基にした仮想ターゲットを用いた卓球タスク

仮想ターゲットに基づく主要なストローク動作の生成には Koditschek [7] が提案したミラー則（mirror law）に類似の視覚フィードバック制御を用い，飛んで来るボールに対して仮想的なミラーを予想打撃位置に設定し，そのボールの鏡像の位置を追従するようにラケットを制御する．こうすることで，ボールが仮想ミラーの位置に到達するときに自動的にラケットはボールを捉えるので，打撃時刻を正確に予測する必要なく打撃が実現できる．ただし，単純に鏡像を追従すると必ずボールと同じ速度で打撃することになり，返球位置を制御することは困難であるため，パラメータ (ミラーゲインと呼ぶ) を用いてミラーから鏡像までの距離を調整し，適切な速度が実現されるようにした．

このときの，ミラーゲインと仮想ミラーの設置位置を仮想ターゲットとして，k dimensional tree(k-d tree) と呼ばれるデータ構造を利用した入出力マップ [8, 9] を用いて予測する．この方法は Atkeson に提案された” タスクレベルロボット学習” の考え方にヒントを得たものである [10]．また，これらの提案手法を用いて行った卓球タスクの実施結果を示す．

1.3　学習制御とマップを用いた卓球タスク

上に述べた研究は次の二つの問題を提示している．一つは，提案されたフィードバック体系ではボールの返球軌道の高さをコントロールすることができない．その場合，ボールがネットを飛び越えられない可能性がある．もう一つは，関節サーボによるトラッキング・エラーにより，制御体系が正確に動作できないということである．トラッキング・エラーが無視できなければ，サーボの制御問題を軌道の計画問題と分けて考えることはできない．

ここでは，設定した飛行軌道に沿ってボールをテーブル上の指定した位置に打ち返すためのラケットの制御方法を提案した．提案した手法は，局所重み付き回帰 (LWR) を利用した次の三つの入力出力マップを含んでいる [11]:

(1) 飛来ボールの状態を記述する入力ベクトルによって，ラケットがボールを打撃する時刻とその時のボールの位置および速度を予測するマップ

(2) 打撃直前直後のボール速度の変化を示すマップ

(3) 打撃直後のボールの速度と返球したボールの跳ねる位置とおよび飛行時間の関係を表す逆マップ

これらのマップは，上述のラケット制御に用いる仮想ターゲットを予測するために導入した．3 番目のマップは，Andersson が考慮しなかった理想位置へのボールを打ち返しを実現する．これらのマップを局所重み付き回帰 (LWR)[11] によってシステムに組み込んだ．他に，マルチレイヤ・ニューラルネットワーク (NN) やラジアル基本関数 (RBF) などのような選択肢も考えられ，効率的なオンライン学習アルゴリズムも提案されてはいるが [12]，通常の NN では，収束率が遅く，我々の目的としている問題には実用的でない．

RBF はより高速な学習に適しているが，信頼性の低い知覚データを排除せずに RBF を適用すれば，RBF の入出力関係の学習能力が悪化する恐れがある．RBF と比べて，同様に高速な学習に適した LWR は，信頼性の低いトレーニングデータを明確に発見する能力を容易に組み込むことができる．さらに，Gorinevsky と Connoly はノイズを加えたロボットシミュレーションの逆運動学によっていくつかの近似手法 (NN,RBF,LWR) を比較し，LWR が他の手法より正確であることを示した [13]．

ひとたび，ラケットの軌道計画が生成されれば，次にはそれをできる限り正確に実現する必要がある．そこで，ロボット制御装置のサーボ遅れやシステムの弾性要素による制御の誤差を補償するために，我々は反復学習制御 (ILC)[14] に基づくフィードバック制御システムを提案する．この提案手法は，適切な動作を実現できるように既に正確に学習された入力コマンドを線形に組み合わせることにより，繰返し学習を行わずに新しい動作軌道が実現できるような入力信号を生成する．

要点をまとめると，以下のようになる．

① ロボットと環境との断続的な相互作用下における，ロボットのタスクに含まれているハイブリッド制御問題に対して，メモリベースド学習手法の有効性を証明する．

② 順マップと逆マップの組合せにより，ロボットシステムが動的な環境で適切な動作計画が可能であることを実証する．

③ 入出力マップを用いて作成された動作計画を，正確に実現できるフィードフォワード制御の手法を提案する．

1.4 人間の打撃動作の計測と Master-Slave を用いた学習システム

人がスポーツを始める際，まずは基本的なフォームを身につける．卓球においても同様である．これは，打撃動作はある一定のパターンによって行われるためと考えられる．ある特定のパターンのボールが来た場合ならば確実に返球出来るような，基本フォームとなる動作を身につけておき，新たなボールに対しては，その基本のボールパターンとの相違を考慮して適切に打撃を調整すると考えられる．人の動作をもとに動作の基本パターンを作成し，アトラクタの形を用いて運動の類似性を認識することによって，模倣学習を行う研究がなされている [15]．他にも，模倣を取り上げた研究は数多く存在するが，それらのほとんどは空間的な運動の模倣であり，時間には考慮していない [16],[17]．我々は運動そのものの模倣学習を一段進めて，外界の環境変化に対する対応能力の学習というところに焦点を当てて研究する．

環境変化を表す対象物体 (卓球タスクにおけるボール) の運動を制御対象となるロボットの運動にマッピングすることを目的として，手先や身体の位置および速度情報から，ボールの運動に対する人の動作パターンの傾向を取得し，運動を記述する方程式を学習制御を行うための単位に分解する．その中で，ボールの運動と関連性のある動作とそうでない動作を分類し，基本動作と環境変化への対応動作を別々に扱うことによって，ロボットへの対応能力の適用を目指す．

人間の動作をロボットの制御に適用するために，人間の動作の計測を行い，人のデータを用いて予測や行動決定を行う手法を考察した．ラケットスイングの計測には磁気を利用した 6 次元の計測が可能な FASTRAK を用い，打ち返しの目標位置の違い，打ち返すボールの速度，飛んで来るボールの速度，飛んで来るボールの位置などの変化によるスイングの傾向を調べた．その結果をもとに，以下のような戦略で人がタスクを実現する際のタイミングや動作パターンおよびボールの挙動を，ボールの速度や位置などの情報とやタイミングや待機位置との関係を表すマップとして直接ロボットが学習しながらダイナミックマニピュレーションタスクを実現する．

人間の打撃動作の計測により卓球の打撃動作パターンは「飛来するボールに対応した打撃動作のための位置調整」と「ある程度決まったボールとの相対位置からの一定の打撃動作」の大きく 2 つに分けられると推測される．そこでまず，人の動作によって直接ロボットを操作する Master-Slave 方式を用いて人間が打撃タスクを行う際の一定の打撃パターンを抽出する．次に，Master-Slave 方式による「位置調整」(バックスイング) 動作から，抽出した「一定の打撃動作」への切替えを含めた打撃実験を行う．このとき人間が，一定動作で打撃が行えるような切替えのタイミングをリアルタイムに与え，飛来するボールと切替えの位置やタイミングとの関係をマップとして学習する．

最後に，そのマップを用いてバックスイングと一定動作への切替えを予測，実行してロボットによる自律的な打撃動作を実現する．

1.5　本研究の目的と本論文の構成

ロボットをより環境に適応させるために，環境から受ける感覚としての知覚情報を処理し，運動という形で再び環境に影響を与える際のプロセスをシステム化することを目的とし，人が外界の刺激と運動を関連づけることで実現する動的な環境への対応能力を要するタスクとして，卓球を取り上げて研究を行ってきた．人間は変化するターゲットへの対応を熟練によってこなし，更にその行動を繰り返すことで無駄なくスムーズに作業をこなすことができるようになる．この点に注目して，本研究では人の動作とターゲットの変化に対する対応の手段をロボットに適用することにより，ダイナミックに変化するターゲットへの適応能力を開発することを目的とする．

最後に，本論文の構成を示す．第 2 章では，本研究に用いたシステムの紹介を行う．卓球ロボット P2 について，その制御用機器，駆動システムや動作範囲，性能などを説明する．また，環境認知，具体的には主にボールの位置計測に用いる画像処理システム QuickMAG の概要を述べる．さらに，人間の動作の計測においてラケット位置・姿勢を計測する 6DOF

磁気計測システム (FASTRAK) について説明し，FASTRAK を用いた Master-Slave システムも紹介する．

第 3 章では，卓球タスクを予測，打撃動作，ボール監視の 3 つのタスクに分け，Koditschek らが提案したミラー則を応用した打撃動作の生成手法について述べる．仮想ターゲットとして打撃位置とラケット速度を調整するパラメータを k-d tree 構造のマップを用いて予測し，返球位置を指定してボールの打撃を行った結果を示す．

第 4 章では，サーボ誤差やシステムの弾性の問題から生じる，目標の軌道に対する実機の遅れを解決するためのフィードフォワード制御として学習制御について述べる．通常の学習制御を応用した，卓球タスクにおける動的に変化する目標軌道に対しても繰り返し学習を必要とせずに適切な動作を実現する入力コマンド列を生成する Direct ILC 手法について述べる．

第 5 章では，卓球タスクにおけるボールの物理現象を 3 つのマップによって記述し，LWR を用いてこのマップから予測された仮想ターゲットを，上記の Direct ILC によって正確に実現することによりボール操作タスクを行う．ボール操作タスクは，飛来するボールに対し，その飛距離と返球位置を自由に指定して打撃を行うタスクである．さらに，同様の手法で実現した対人のラリータスクについても紹介する．

第 6 章では，人間の動的タスクに対するスキルを理解するため，人間が卓球タスクを行う際の動作を計測し解析した．供給するボールや返球の位置，速度などを変化させて，それぞれの打撃動作について調べ，その時の学習の成果やスキルをロボットへ適用するための手法を考察した．

第 7 章人間の打撃動作の解析結果から得られた，より直接的な人間のダイナミックマニピュレーションスキルのロボットへの適用手法を提案する．ボールの打撃動作をボールに合わせて調整するバックスイングと一定パターンの打撃動作の 2 つに分け，まずはこのパターンやこれを切替えるタイミングを直接人間の動作から抽出する Master-Slave システムについて述べる．さらに，動作の切替えタイミングと位置を LWR 入出力マップを用いて予測し，Direct ILC と一定パターンへの切替えを用い，ロボットが自律的に打撃動作を行った結果について述べる．

第 8 章に本研究のまとめを行うとともに，今後の課題について述べる．

第 2章　システム構成

本研究では卓球タスクを取り上げて，ロボットの動的対応能力の研究を行っている．本節では，本研究で用いる卓球ロボットシステムの構成について述べる．

2.1　基本ロボットシステムの構成

卓球ロボットは**図 2.1** のような構造をしている．卓球タスクを実行するには最低限，ラケットの姿勢 2 自由度とラケットの前後・左右方向の位置 2 自由度の合計 4 自由度が操作できればよい．そのため，現在の卓球ロボットは，モーター 1,2 によってラケットの前後 (X), 左右 (Y) 方向の動作を，モータ 3,4 によってラケットの姿勢角度を生成することができる．各モータの可動範囲は以下のように設定した．

Motor 1 [-1100 [mm] $\leqq X \leqq$ -350 [mm]]

Motor 2 [-400 [mm] $\leqq Y \leqq$ 400 [mm]]

Motor 3 [-90 [deg] $\leqq \theta_3 \leqq$ 90 [deg]]

Motor 4 [-60 [deg] $\leqq \theta_4 \leqq$ 60 [deg]]

また，速度の上限は $|V_{rx_{max}}|, |V_{ry_{max}}| = 6000$[mm/s] とした．

全体のシステム構成を**図 2.2** に示す．2 台の CCD カメラで捉えたボール画像は QuickMAG で解析され，60[Hz] の sampling rate でボールの位置データが PC に送られる．PC によって動作計画やモータへの指令生成を行い，pulse generator を介してモータードライバに速度指令を入力することでラケットの打撃動作を実行する．

2.2　計測システム

人のスイングやボールなどの環境計測のため，以下の 2 点の計測機器を用いる．

- QuickMAG(OKK 社) → ボール位置計測用画像処理計測システム
- 3SPACE FASTRAK(日商エレクトロニクス社) → ラケット位置&角度計測用磁気計測システム

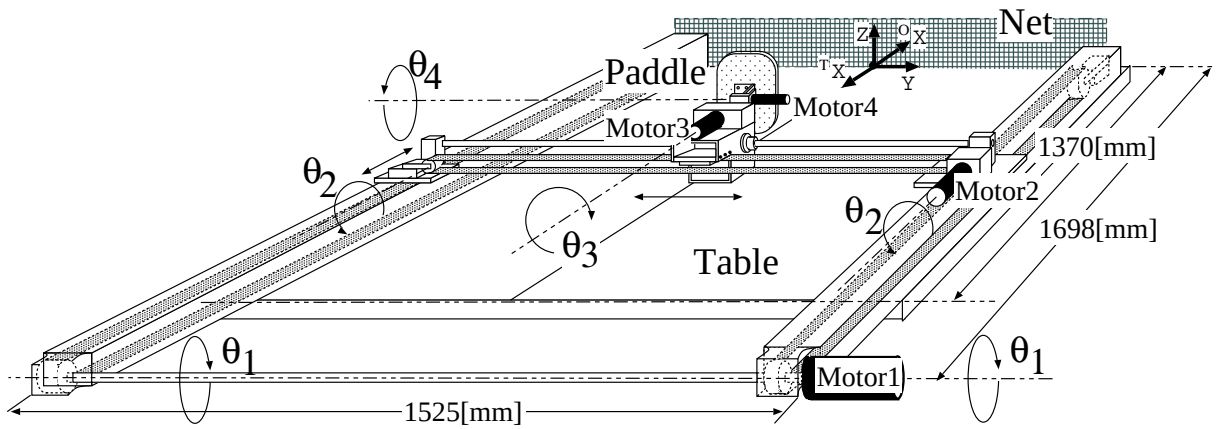


図 2.1　Table tennis robot system

2.2.1　視覚処理システム

ボールの 3 次元位置を計測するためにリアルタイム動作解析システム (応用計測研究所製 Quick Mag System III) を使用する．QuickMAG は物体の色を認識し，左右 2 台のカメラ画像から物体の対応点を抽出し，その 3 次元位置を DIO ポートから 60Hz のサンプリングレートで制御用コンピュータへ出力する．QuickMAG の仕様を**表 2.1** に示す.

表 2.1　QuickMAG 本体仕様

画像入力	RGB
画像出力	RGB
複合同期出力	VBS
ステータス出力	8 ビット TTL
入出力 I/F	GP-IB
デジタル出力	8 ビットパラレル I/O にて座標データ出力
物体検出能力	任意色同時 8 色抽出
全画面座標	736×480
有効範囲	640 × 416
測定周期	1/60 秒
測定最大窓数	(2 次元)16，(3 次元)8
窓寸法	最大 640 × 416
窓形状	矩形

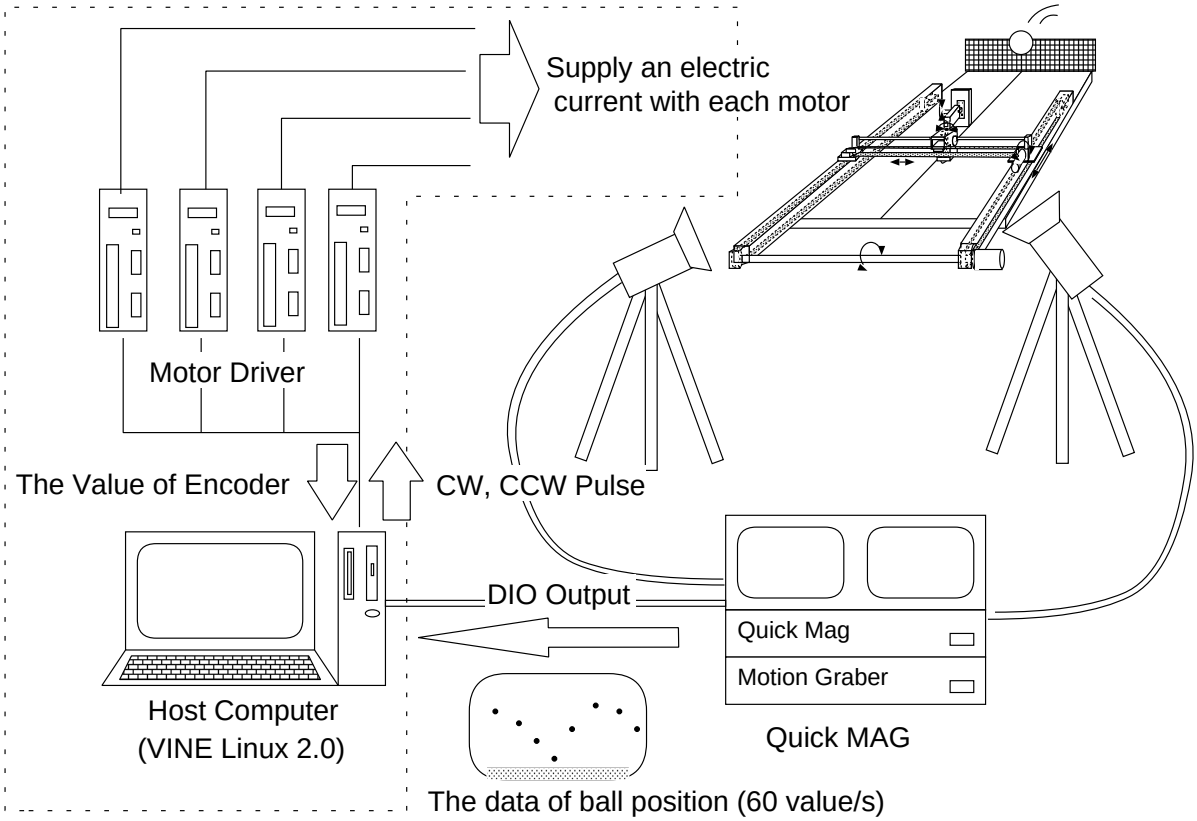


図 2.2 System configuration

2.2.2 磁気計測システム (3SPACE FASTRAK)

人間の肘とラケットの 3 次元位置を計測するために高精度 3 次元位置センサー 3SPACE FASTRAK(日商エレクトロニクス) を使用する．FASTRACK は磁気変換技術を用いて，3 次元位置座標値 (X,Y,Z) および、オイラー角 (Pitch, Yaw, Roll) の 6 自由度をリアルタイムに測定することができる．FASTRAK の仕様は**表 2.2** に示す通りであり，**図 2.3** に計測範囲を図示した．卓球のラリーにおけるラケット角度はおおむねこの範囲内で行われている．レシーバーは肘およびラケットの 2 箇所に用いているので，60[ポイント/秒] で計測される．

2.3 人間の動作計測環境

FASTRAK のセンサーは**図 2.4** に示した位置に取り付けた．センサー 1 はラケットの打撃部分の裏側中央に，また，センサー 2 は肘用のサポーターに縫いつけて，肘間接の位置に来るように装着した．

アームの角度は次のように求める．まずアームの長さを 2 つのセンサー間の距離として，3 次元のアームの長さ $L_{3,arm}$ と，X-Y 平面上に投影したアーム長さ $L_{2,arm}$ を求める．

$$L_{3,arm} = \sqrt{l_x^2 + l_y^2 + l_z^2} \tag{2.1}$$

表 2.2　FASTRAK 本体仕様

測定自由度	6DOF (x, y, z, Pitch, Yaw, Roll)
精度	位置：0.8mm RMS、角度 0.15 度 RMS
測定範囲	半径約 106cm の半球内
レシーバ (センサー) 数	最大 4 レシーバ
データレート	1 レシーバ使用時：120 ポイント／秒 2 レシーバ使用時：60 ポイント／秒 3 レシーバ使用時：40 ポイント／秒 4 レシーバ使用時：30 ポイント／秒
インタフェース	シリアル（RS-232C）接続
外部同期ポート	SYNC IN × 1、SYNC OUT × 1
データ形式	アスキー／バイナリ　切替可

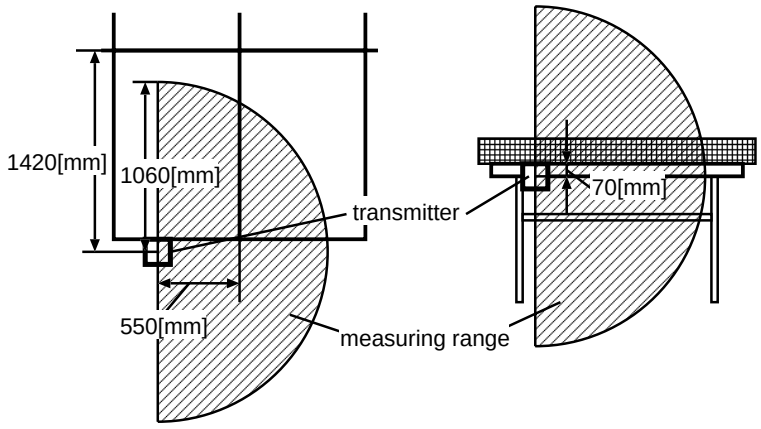


図 2.3　FASTRAK の計測範囲

$$L_{2,arm} \quad = \quad \sqrt{l_x^2 + l_y^2} \tag{2.2}$$

ただし，センサー 1, 2 の座標をそれぞれ $(x_1, y_1, z_1), (x_2, y_2, z_2)$ とするとき，$l_x = x_1 - x_2, l_y = y_1 - y_2, l_z = z_1 - z_2$ である．アームの roll 角 α_{arm} はアームの長さ $L_{3,arm}$ とその z 成分ですぐに求まる．

$$\alpha_{arm} = -\sin^{-1}(\frac{l_z}{L_{3,arm}}) \tag{2.3}$$

yaw 角 β_{arm} は，$L_{2,arm} \simeq 0$ の時はラケットの yaw 角 β_{paddle} と等しいとして，センサー 1 の値をそのまま用いる．$L_{2,arm}$ が 0 でないとき，$l_y > 0$ の場合，$l_y < 0$ かつ $l_x > 0$ の場合，

$l_y < 0$ かつ $l_x < 0$ の場合の 3 つに場合分けして求まる．

$$\beta_{arm} = \begin{cases} \beta_{paddle} & ,(L_{2,arm} \simeq 0) \\ \cos^{-1}(\frac{l_x}{L_{2,arm}}) & ,(l_y > 0) \\ \sin^{-1}(\frac{l_y}{L_{2,arm}}) & ,(l_y < 0 \text{ かつ } l_x \geqq 0) \\ -\pi - \sin^{-1}(\frac{l_y}{L_{2,arm}}) & ,(l_y < 0 \text{ かつ } l_x < 0) \end{cases} \tag{2.4}$$

pitch 角 γ_{arm} はラケットの pitch 角 γ_{paddle} と等しいと考え，センサー 1 の値をそのまま用いる．

図 2.5 に計測環境の全体図を示す．通常の卓球台を用い，ネットは取り付けない．ボールは 40mm のオレンジ色の公式球を用いる．飛来するボールを一定にするために配球は打球機を用いて行い，反対側のコートで人間があらかじめ設定した目標位置に打ち返す．QuickMAG のカメラは打球機の後方から計測する．この環境下で，ボールの位置データおよびラケット位置，角度，肘の位置を計測する．トランスミッターの位置 (つまり FASTRAK の原点) は QuickMAG 座標で計算すると，だいたい $(-1420, 455, -60)$ である．

2.4 Master-Slave システム環境

また，Master-Slave によるロボットの制御システムを**図 2.7** に図示する．この環境下で，ボールの位置データおよびラケット位置，角度，肘の位置を計測する．人とロボットの後方に立ち，反対側の集球ネットの裏に設置した打球機から打ち出されるボールに対してスイングを行って操作し，ロボットが設定した目標位置への打撃を行う．ボールは 40mm のオレンジ色の公式球を用い，QuickMAG のカメラは打球機の後方から計測する．FASTRAK のトランスミッターの位置 (つまり FASTRAK の原点) は QuickMAG 座標で計算すると，およそ $(1620, -455, -60)$ である．

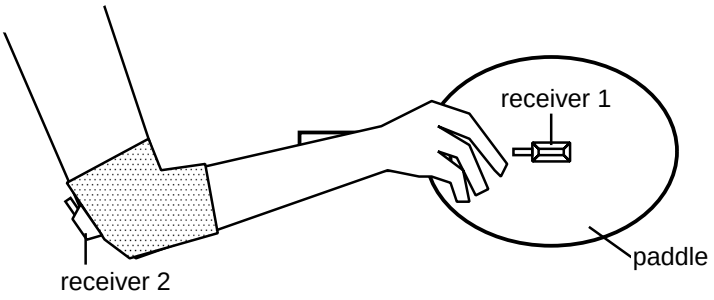


図 2.4 FASTRAK センサーの取付位置

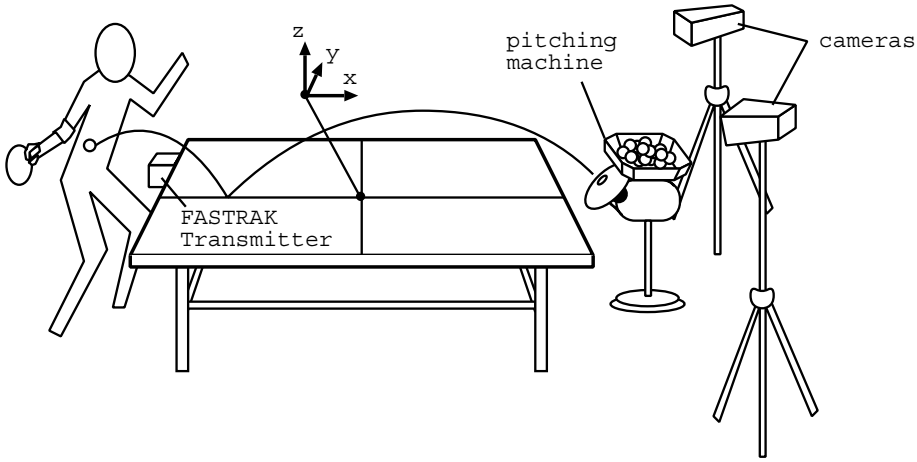


図 2.5　人の打撃動作計測システム

手順 1(人間のラケットの計測)　**図 2.4** に示すように，Fasktrak センサーが人間のラケットの中心に貼りつけられている．その座標系は，**図 2.6**(a) のロボットの座標系と異なり，**図 2.6**(b) に示すような ${}^fX$,${}^fY$,${}^fZ$ 軸を取る．${}^fY$ 軸が面に対して垂直方向の軸である．また，角度の座標系は**図 2.6**(c) に示すように，${}^fX$,${}^fY$,${}^fX$ 軸まわりにそれぞれ roll,pitch,yaw という名前をつける．人間のラケットの生データがこの座標系で計測される．

手順 2 Fastrak からロボットへの座標変換　位置のデータの変換には，次式のような単純な変換を行う．

$$\left\{ \begin{array}{l} {}^fX \rightarrow {}^rY \\ {}^fY \rightarrow {}^rX \\ {}^fZ \rightarrow -{}^rZ \end{array} \right. \Longleftrightarrow \left(\begin{array}{c} {}^rX \\ {}^rY \\ {}^rZ \end{array} \right) = \left(\begin{array}{ccc} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & -1 \end{array} \right) \left(\begin{array}{c} {}^fX \\ {}^fY \\ {}^fZ \end{array} \right) \tag{2.5}$$

次に，角度の変換であるが，以下のように座標変換行列を求めた．

それぞれの回転順序は，ロボット座標系においては X → Y オイラー角，Fastrak のセンサー (角度) に関しては，Y→X→Z オイラー角となっている．従って，目標角度 θ_3, θ_4 は，Fastrak の X,Y,Z 軸回りの角度 (α, β, γ) から，次のように逆運動学を計算した．ロボット座標からワールド座標系への回転行列は，

$${}^oR_r = \left(\begin{array}{ccc} \cos\theta_4 & 0 & \sin\theta_4 \\ \sin\theta_4 \sin\theta_3 & \cos\theta_3 & -\cos\theta_4 \sin\theta_3 \\ -\sin\theta_4 \cos\theta_3 & \sin\theta_3 & \cos\theta_4 \cos\theta_3 \end{array} \right)$$

これに対して，人のラケット (Fastrak) 座標からワールド座標への変換行列は，X，Y，Z

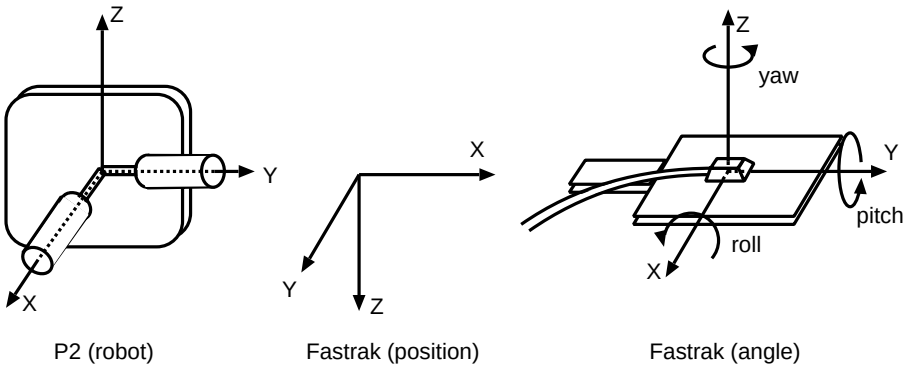


図 2.6 Fastrak&ロボット (P2) の座標系

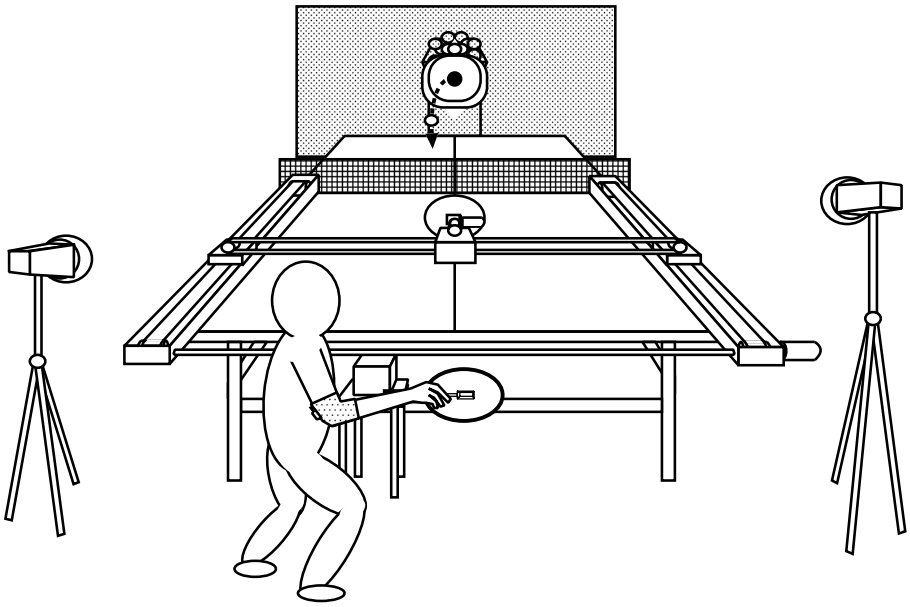


図 2.7 Master-Slave システム

各軸回りの回転を r(=roll), p(=pitch), y(=yaw) とし，sin→s, cos→c と表記すると，

$$
\begin{aligned}
{}^{o}R_{f} &= \begin{pmatrix} cp & 0 & sp \\ 0 & 1 & 0 \\ -sp & 0 & cp \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 0 & cy & -sy \\ 0 & sy & cy \end{pmatrix} \begin{pmatrix} cr & -sr & 0 \\ sr & cr & 0 \\ 0 & 0 & 1 \end{pmatrix} \\
&= \begin{pmatrix} cpcy + spsrsy & -cpsy + spsrcy & spcr \\ crsy & crcy & -sr \\ -spcy + cpsrsy & spsy + cpsrcy & cpcr \end{pmatrix} \qquad (2.6)
\end{aligned}
$$

(2.7)

ここで，Fastrak の角度の座標系の Z 座標とロボット座標系の X 座標が一致すればいいので，

$$\cos\theta_4 = spcr \qquad (2.8)$$

$$\sin\theta_4 \sin\theta_3 = -sr \tag{2.9}$$
$$-\sin\theta_4 \cos\theta_3 = cpcr \tag{2.10}$$

よって，求めるモータの角度 θ_3, θ_4 は，

$$\theta_4 = \begin{cases} \cos^{-1}(spcr) & (cpcr < 0) \\ -\cos^{-1}(spcr) & (cpcr \geqq 0) \end{cases} \tag{2.11}$$
$$\theta_3 = Atan2(-\frac{sr}{\sin\theta_4}, -\frac{cpcr}{\sin\theta_4}) \tag{2.12}$$

となる．θ_4 の場合分けは $\cos^{-1}()$ の戻り値が正の数のみなので，$-180[deg] \leqq \theta_4 \leqq 180[deg]$ とするためである．

しかし，ラケットの追従テストを行ったところ，**図 2.8** に示すように問題が発生した．ラケットの面を右上 $\leftrightarrow$ 右下，右上 $\rightarrow$ 左上，左上 $\leftrightarrow$ 左下の順に動かしたところ，右上 $\leftrightarrow$ 右下および左上 $\leftrightarrow$ 左下の移動の際に，θ_3 および θ_4 の角度が正負反転し，この際に，速度の最大値を越えてしまうため追従ができなかった．

これは，卓球システムの構造上の問題であり，両方の角度を追従することには限界があると考えられる．そこで左右方向の角度の追従を諦め，人のラケットの上下方向のみ追従することとし，ボールも X 軸方向のみの動作に限定することにした．つまり，角度の変換は以下のようになる．

$$\theta_4 = r(=\text{roll}) - 90[deg] \tag{2.13}$$
$$\theta_3 = 0 \tag{2.14}$$

手順 3 速度コマンドの生成　手順 1,2 で求められたパドルの目標位置/速度と現在位置/速度に応じて，以下のようにロボットに与えられる速度コマンドが生成される．

$$v_x = K_p * ({}^{*d}p_x - {}^{p2}x) + K_d * ({}^{*d}v_x - {}^{p2}v_x) \tag{2.15}$$
$$v_y = K_p * ({}^{*d}p_y - {}^{p2}y) + K_d * ({}^{*d}v_y - {}^{p2}v_y) \tag{2.16}$$
$$\omega_3 = 0 \tag{2.17}$$
$$\omega_4 = K2_p * ({}^{*d}\theta_4 - {}^{p2}\theta_4) \tag{2.18}$$

ここで，目標値 ^{*d} は最小二乗法による推定値，^{p2} はエンコーダからの実際の値を表す．位置と速度のフィードバックゲインはそれぞれ，$K_p = 3.0, K2_p = 10.0, K_d = 1.0$ とした．ただし，加速度一定 $({}^{*d}p_x = {}^{d}a_x\delta t^2 + {}^{d}v_x\delta t + dp_x)$ として最小二乗近似を X, Y 両軸について行った．(Δt:最後に Fastrak を実測してから現在までの時間，${}^{d}v_x$:最後に実測した時の Fastrak の速度，${}^{d}p_x$:最後に実測した Fastrak の位置)．

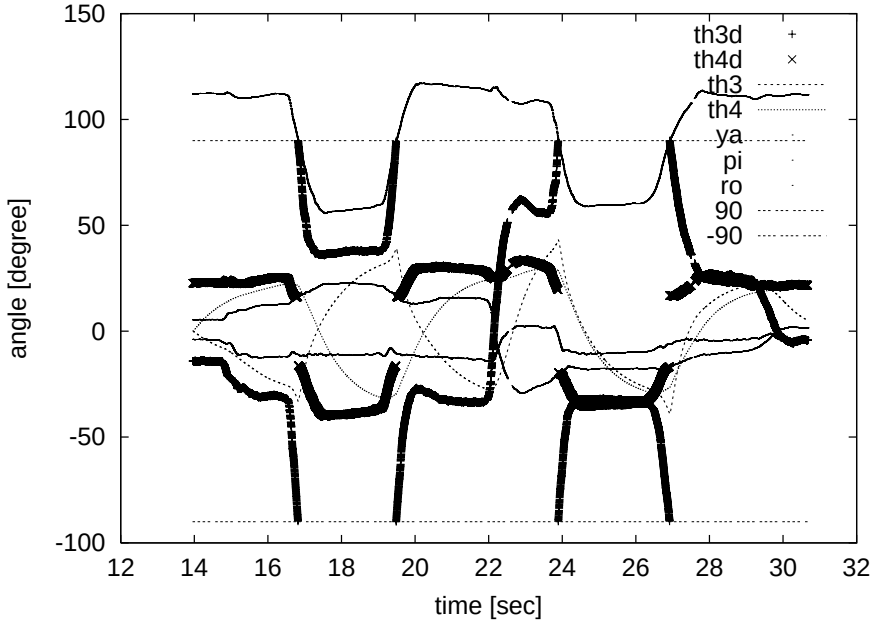


図 2.8 角度の追従 (θ_3 の問題)

第 3章　仮想ターゲットの予測に基づく卓球タスクの実現

3.1　はじめに

ダイナミックに変化する環境下でロボットと環境が断続的に相互干渉するようなタスクを実行する場合，ロボットには連続動作しながら相互干渉の強さやタイミングを調整する機能が要求される．このような問題を Burridge ら [18] はハイブリッド制御問題と呼んでおり，タスク例としてダイナミックな歩行や走行，ボールのスローイングやキャッチングなどをあげている [7, 19, 20]．人間の身体運動を解析する分野でも，ターゲットにタイミングを合わせて行うこれらの動作をタイミング動作と呼び，巧みさを感じさせる動作として注目している [21]．しかし，ハイブリッド制御問題に対する一般的なアプローチと言えるものはなく，タスクに応じて個別の手法が提案されているのが現状である．本論文でとり上げる卓球タスクも典型的なハイブリッド制御問題であり，Andersson や橋本らがロボットによる実行結果を報告している [5, 22]．Andersson は，ボールやロボットの動特性の陽なモデルを利用してボールの打撃動作生成の問題をボールの状態予測とラケットの軌道生成の問題に分離し，ボールの状態の逐次予測とそれに基づくラケットの目標軌道の更新によってハイブリッド制御問題に対処する方法を示した．しかし，ロボットが生成できるラケット速度や加速度に限界があるため，実際にはヒューリスティックに設定した例外処理を多用している．

一方，スポーツ科学の分野では，トッププレーヤのラケット操作の解析を通し，ボールを打ち返してから再び飛来するボールを打つまでの１ストロークのラケットの動きについて様々な仮説が示されている．その中で Ramanantsoa は，１ストロークの動作を４つのフェーズに分割し，各フェーズごとにボールの動きに合わせて時空間的な動作調整を行うモデルを提案している [6]．このモデルでは，ボール打撃時のラケットスピードや位置を仮想ターゲットと呼び，ストローク動作はこの仮想ターゲットの予測と微調整に基づいて実行されるものと考えている．

本章ではこのような考え方をロボットによって具体化する手法について述べる．仮想ターゲットの予測には k dimensional tree(k-d tree) と呼ばれるデータ構造を利用した入出力マップを，仮想ターゲットに基づく主要なストローク動作の生成には Koditschek が提案したミラー則（mirror law）に類似の視覚フィードバック制御を用いる．

仮想ターゲットの１つである打撃位置の予測は，飛来するボールのスピードやコースの観測結果からボールの打撃位置をできるだけ早めに決定する問題であり，ボールの空力特性や卓球台との反発特性を十分考慮する必要がある．本論文では，飛来するボールの状態を入力，打撃位置を出力とした入出力マップを利用して打撃地点を予測する方法を提案す

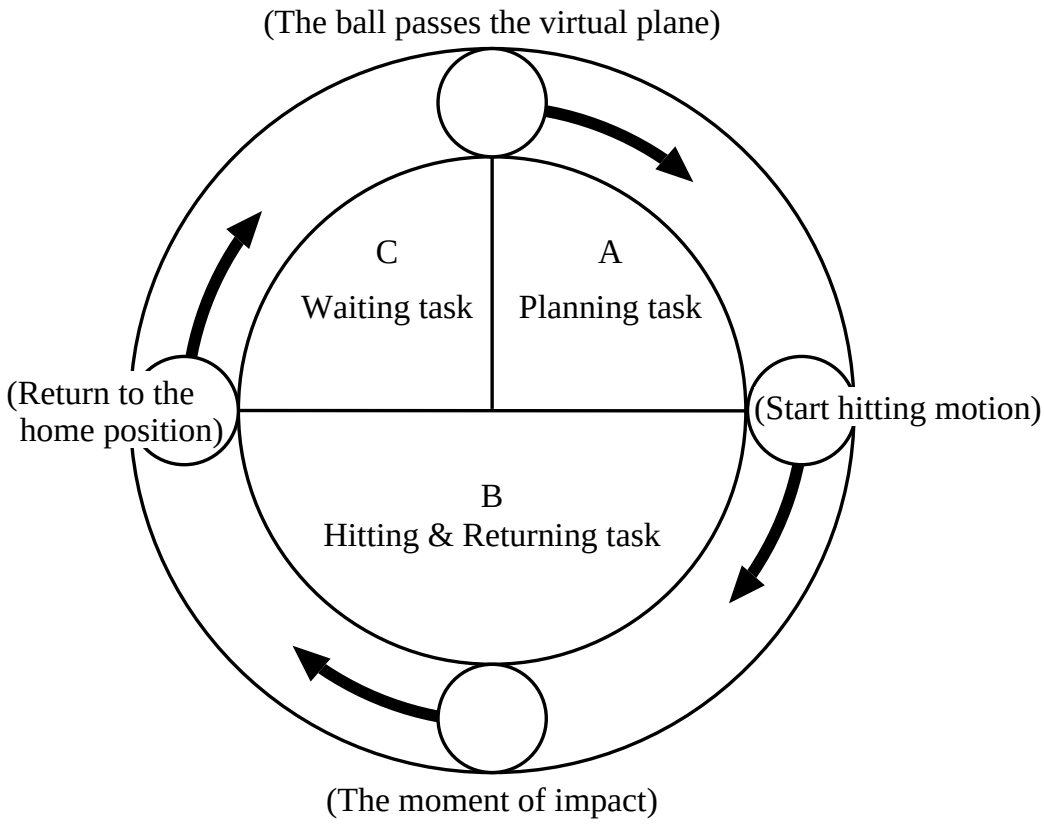


図 3.1　Task oriented description of stroke movement

る．もう１つの仮想ターゲットである打撃時のラケットスピードの決定は，ボールとラケットの反発特性も加わって打撃位置の予測以上に複雑な物理現象が絡む問題である．本章では，ミラー則に類似した制御則によってラケットを駆動するものとし，ラケットスピードに対応する制御パラメータを打撃位置の予測と同様の入出力マップを利用して決定する方法を提案する．また，これらの提案手法を用いて行った卓球タスクの実施結果を示す.

3.2　卓球タスクの実現方法

卓球タスク実現のための基本的な枠組を以下に述べる.

3.2.1　ストローク動作の分割

ボールが飛来してから打撃を行い，次のボールが飛来するまでの一連動作を一つのタスクとしてとらえ，時間の経過を考慮して **図 3.1** のように表現した．時計の 0 時の位置から順に,

TASK A: 打撃位置の予測および動作計画タスク.

TASK B: 打撃および帰還タスク.

TASK C: ボールを監視し続け，TASK A への移行タイミングを調整する.

TASK A は，ボールを監視し，予測に基づいて打撃動作を行うため仮想ターゲットを生成するタスクである．TASK B は飛来するボールを打ち返す，卓球タスクの核となるタスクであり，ロボットが TASK A での計画に基づいて打撃動作を行った後，待機位置へ帰還する．C は，ボールが再び飛来するまで待機するタスクであり，動作を伴わないがボールの状態を常時観測しながら仮想ターゲットを決定し A のタスク開始のタイミングを調整する役割を担う.

以下では，卓球タスクの中で主要な動作をともなう B のタスクの実現方法についてその概要を説明する．なお仮想ターゲットの決定方法 (TASK A) については，その次に詳しく述べる.

3.2.2 ボールの打ち返し動作の生成方法

ここでは，既に仮想ターゲットが予測/動作計画タスク (TASK A) において決定されているものとし，決定されたボール打撃時のラケットスピードおよび位置を実現する動作の生成方法について説明する．仮想ターゲットを実現するためにラケットの軌道を Andersson のように時間関数として与えた場合，打撃時刻を陽に求める必要がある．予測した仮想ターゲットに誤差があれば打撃時刻も誤差を生じるため，予測誤差の影響は拡大される．このため，仮想ターゲットの逐次更新は避けられない．このような方法に対し，本章では打撃時刻を求める必要のない方法を提案する．**図 3.2** は，飛来するボールの位置に応じてラケットの位置を変化させ，予測した打撃位置でボールを打ち返す一連のラケット動作を 2 次元的に表現したものであり，卓球台中央のネットの法線方向に X 軸，鉛直方向に Z 軸をとっている．提案手法では，ラケットは高さ一定の水平面内を移動するものとし，その X 方向の目標位置 X_p を以下のように与える（XZ 平面に垂直な Y 軸方向の移動も同様の方法を用いる）.

$$X_p = X_m + k(X_m - X_b) \tag{3.1}$$

X_m は予測したボール打撃位置，X_b は時々刻々と変化するボールの位置を表している．(3.1) は，ジャグリングタスクにおける鉛直方向のボール打撃動作生成法として Koditschek ら [7] が提案したミラー則（mirror law）を水平方向に焼き直したものとも言える．すなわち，ボール打撃位置 X_m に鏡を置き，その鏡に写ったボールの像 X_p を追ってラケットを移動させることに対応している.

図 3.2 において、ボールが①にある時はミラーに写った像①を，②にある時は像②をラケットに追従させることによりボールがミラー位置 X_m 到達時 (③) にはラケットもミラー位置に達するため，ミラー位置 X_m でボールを捉えることができる．なお，(3.1) の k は鏡に写ったボールの像の X 方向距離を調整するパラメータであり，ミラーゲインと呼ぶ．ミラーゲインを調整すればボール打撃時のラケットスピードを変化させることができるため，

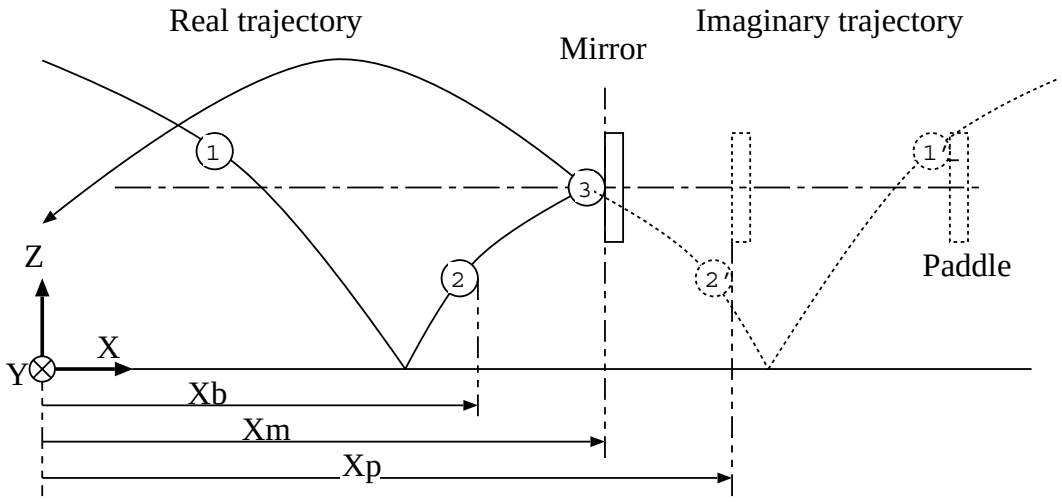


図 3.2　Mirror Hitting

このパラメータを適切に設定することによって仮想ターゲットの実現をはかる．つまり，仮想ターゲットとして打撃位置 (仮想ミラー設定位置)X_m と適切な速度を生成するミラーゲイン k を決定することで，打撃動作の決定を行う．

3.3　仮想ターゲットの決定

ボール打撃時のラケットスピードや位置に対応する仮想ターゲットは，**図 3.1** の予測/動作計画タスク (TASK A) に割り当てられた時間内に決定されなければならない．この許容時間は飛来するボールのスピードに大きく依存し，場合によっては予測/動作計画タスクを設けることが不可能になることもある．ここでは，ボールの動きをとらえる視覚処理系のサンプリング時間（$1/60[s]$）を考慮して TASK A の許容時間をボール計測開始から約 $0.2[s]$ とし，その時間内に仮想ターゲットの決定を行うものとする．なお，この許容時間に対応したボールスピードの上限は約 $5[m/s]$ （通常のラリーが続くスピード）に相当する．

3.3.1　打撃位置の決定

図 3.2 に示したボール打撃時のラケット位置 X_m の決定方法について説明する．飛来するボールの動きを **図 3.2** と同様 2 次元的に示した**図 3.3** において，計測開始地点 $X = X_o$ からボール位置 (X, Z) の計測を始めるものとする．ボールが打撃地点に到達するまでに待機タスクと打ち返しタスクを完了しなければならないため，計測開始後取得した数点のボール位置データから打撃位置を予測することになる．この打撃位置は，ボールが卓球台で跳ね返った後にラケットの移動高さ $Z = Z_h$ をよぎる X 方向の位置 X_m であり，計測開始地点付近のボールの位置 (X, Z)，速度 (V_x, V_z)，およびボール自身の Y 軸まわりの回転

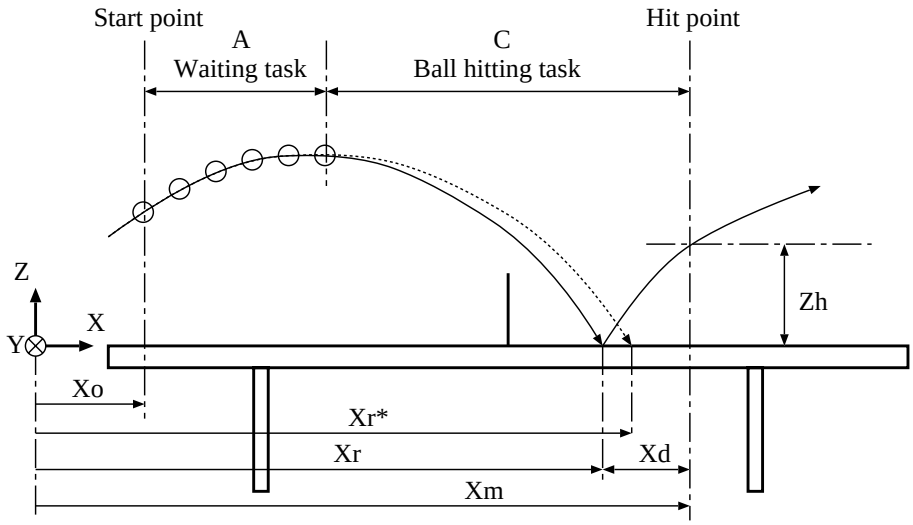


図 3.3 Table Tennis task configuration

角速度 ω_y に依存して変化する．形式的には

$$X_m = f(X, Z, V_x, V_z, \omega_y) \tag{3.2}$$

のような非線形関数として表現できる．この関数を陽に表現するにはボールの空力特性や反発特性を考慮しなければならないが，正確な表現を得ることは困難である．一方，このような非線形関数を入出力マップとみなし，実験的に得られる多くの入出力データを用いてマップを近似表現する方法もある．本論文ではこのような学習的アプローチをとることとし，さらにトレーニングに要する時間やデータの検索時間を短縮するために入力の低次元化を考えた．

ボールが卓球台に衝突するまで角速度の変化がないものとすると，ボールが卓球台と衝突した地点 X_r から $Z = Z_h$ の高さに至るまでの X 方向変位 X_d は

$$X_d = g(V_{xr}, V_{zr}, \omega_y) \tag{3.3}$$

のように，ボールが卓球台と衝突する直前の速度 (V_{xr}, V_{zr}) と角速度 ω_y に依存する．したがって，これらの値と X_r を事前に求めることができれば，低次元化された (3.3) の関係を用いて打撃位置 X_m を決定できる．しかし実際には，ボールが卓球台と衝突する以前に打撃位置を決定しなければならないため，本論文では (3.3) に代えて

$$X_m - X_r^* = g^*(V_{xr}^*, V_{zr}^*, a_z) \tag{3.4}$$

の関係を入出力マップとして学習的に求める．なお (3.4) の X_r^*，(V_{xr}^*, V_{zr}^*) は，計測開始後取得した数点のボール位置データを用いて軌道を時間多項式で最小二乗近似して求めた X_r

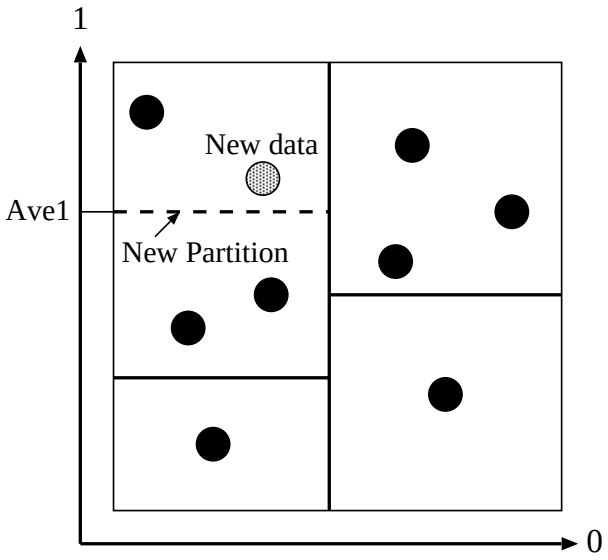


図 3.4　Partitioning of 2-d space

と (V_{xr}, V_{zr}) の予測値である．また角速度 ω_y はボールの速度および加速度の関数と考えられるため [23]，実際には近似軌道の加速度成分 a_z を ω_y に対応したマップの入力とする．

予測を含む入力によって表現される (3.4) の入出力マップが適切な入出力関係を与えるものであることは，3.4.1 節で記述する主成分分析の結果からも裏付けられる．

(3.4) の入出力マップの近似表現方法として，２分木（binary tree）を多次元化した k-d tree を用いる．本研究のように，データ検索に要する時間が厳しく制限され，取得できるデータ量にも制約がある場合に特に効果的な方法と考えられる [10]．k-d tree は，k 次元の入力空間を部分領域にリカーシブに分割し入出力データ対と各領域を対応付けたものであり，２次元の場合を図示すると**図 3.4** のようになる．また，そのときのデータ構造を **図 3.5** に示す．**図 3.4** において，各ノードは入力の成分に対応した判別子（0 or 1）と領域の分割位置を特定する値を持っており，末端ノードには **図 3.5** の各領域内のデータが格納される．この k-d tree を用いて新たな入力データに対する出力を求めるには，まずそのデータが含まれる領域を探索し，領域中にすでに存在する入出力データ対を用いた補間処理を行う．領域中に十分なデータがなければ，近傍の領域も探索の対象に加える．なおトレーニング時には，入力データに対応した領域に入出力データ対を格納していき，格納データ数が事前に設定した上限値を上回る領域があれば，格納データの出力の分散が分割後に小さくなる方向に，その領域を２分する操作を繰り返すことによって領域の細分化を行なう．

3.3.2　打撃時のラケットスピード/ミラーゲインの決定

飛来するボールを卓球台上の目標地点に打ち返すには，ボール速度に応じてラケットの姿勢あるいは打撃時のスピードを調整する必要がある．本論文では，ラケット表面の法線

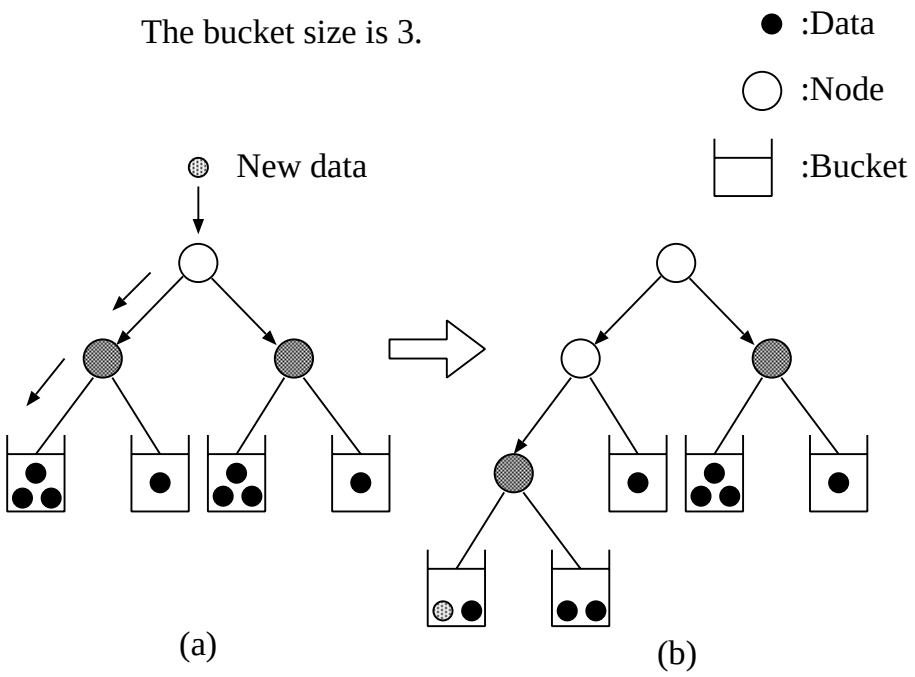


図 3.5 Corresponding k-d tree

が飛来するボールの軌道面 (ボール軌跡を含む鉛直面) に平行し、水平面から一定の角度となるように θ_3,θ_4 を決めて打撃時のラケット姿勢を固定し，X 方向のラケットスピードを制御して目標地点に打ち返すものとする．ラケットで打ち返されるボールの飛距離は打ち返し直後のボール速度および角速度に依存するが，これらの値は打ち返し直前の値とラケットスピードに依存する．さらに飛来するボールが卓球台で跳ね返った後ラケットに衝突するまでの過程を考慮すると，ボールの飛距離 L は

$$L = u(V_{xr}, V_{zr}, \omega_y, V_r) \tag{3.5}$$

のように，ボールが卓球台と衝突する直前の速度 (V_{xr}, V_{zr})，角速度 ω_y およびラケットスピード V_r に依存するものと考えられる．この関係は

$$V_r = w(V_{xr}, V_{zr}, \omega_y, L) \tag{3.6}$$

と書き換えられる．(3.6) はボールが卓球台と衝突する直前の状態と目標飛距離からラケットスピードを決定する関係を表している．実際には，ラケットスピードが打ち返し動作の生成時に用いるミラーゲイン k に対応することを考慮し，さらに打撃位置の決定時と同様に (V_{xr}, V_{zr}) に対してその予測値 (V^*_{xr}, V^*_{zr}) を，ω_y に対して a_z を，L に対して打ち返したボールの着地点 X_g と予測打撃位置 X_m の差をとり

$$k = w^*(V^*_{xr}, V^*_{zr}, a_z, X_m - X_g) \tag{3.7}$$

の関係を入出力マップとして近似的に表現しミラーゲインを決定する．

3.4　仮想ターゲットに基づく打撃実験結果

これまでに述べた卓球タスクの実現方法を実際の卓球ロボットシステムに実装して得られた結果を示す．なおここでは，前節と同じく XZ 平面内の卓球タスク実行結果のみを示すが，実験では，YZ 平面内の卓球タスクも同様の方法で行った．

3.4.1　打撃位置の推定結果

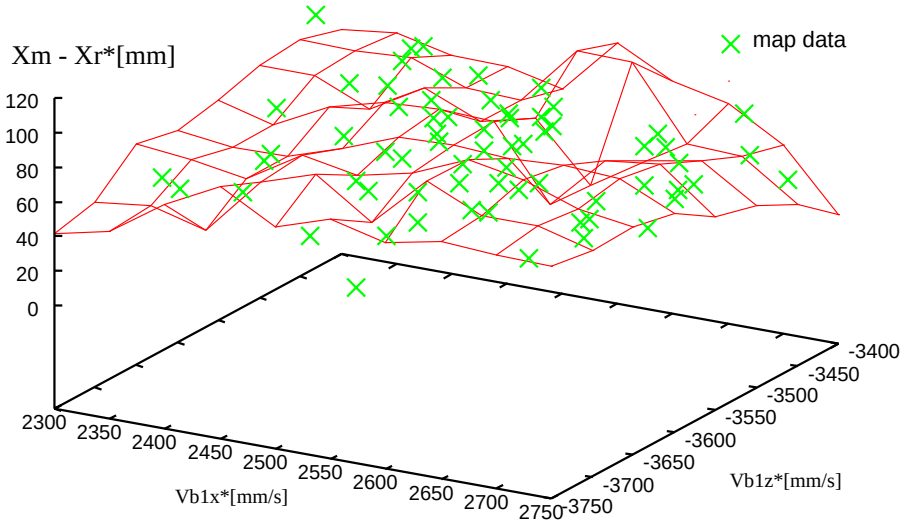


図 3.6　Acquired input-output map

3.3.1 節 節で説明したボール打撃時のラケット位置 X_m の決定方法にしたがって打撃位置を推定した実験結果を示す．まず，実験条件について説明する．**図 3.3** に示した座標系の原点は，卓球台の中央から卓球ロボットの側に 500$[mm]$ 移動した位置にとる．また，相手側卓球台上の $X = -1000[mm]$ の地点からボールの位置計測を開始するものとする．入出力マップ作成時には，この地点を通過した直後の 133$[msec]$ の間に取得される 8 点の位置データから最小 2 乗法を用いて Z 方向の加速度成分 a_z およびボールが卓球台と衝突する位置 X_r^* ，速度 (V_{xr}^*, V_{zr}^*) を推定するとともに，ボールが卓球台と衝突して $Z = 195[mm]$ の高さになったときの X 方向位置 X_m を計測し，入力 $(V_{xr}^*, V_{zr}^*, a_z)$，出力 $(X_m - X_r^*)$ のペアをマップデータに加えていく．

まず最初に，マップデータの相関関係を調べるために行った主成分分析の結果を示す．517 個の入出力変数ベクトル $(V_{xr}^*, V_{zr}^*, a_z, X_m - X_r^*)$ に対する相関係数行列 R は

$$R = \begin{pmatrix} 1 & 0.5632 & -0.0089 & 0.2802 \\ 0.5632 & 1 & 0.5258 & -0.0550 \\ -0.0089 & 0.5258 & 1 & -0.8208 \\ 0.2802 & -0.0550 & -0.8208 & 1 \end{pmatrix} \tag{3.8}$$

となる．この行列 R の固有値，固有ベクトルは**表 3.1** の通りである．固有値 λ_1,λ_2 の累積寄与率 $(\lambda_1+\lambda_2)/\sum_{i=1}^{4}\lambda_i$ が 0.9 となることから，ほとんどのデータが λ_1,λ_2 に対応した固有ベクトルで張られる平面付近に分布していることがわかる．また，λ_1 に対応した第一主成分 z_1 は

$$z_1 \simeq 0.439V_{zr}^* + 0.692a_z - 0.567(X_m - X_r^*) \tag{3.9}$$

より，スピン等の影響を大きく受けるボールの Z 方向の運動状態と飛距離の関係を，一方 λ_2 に対応した第二主成分 z_2 は

$$z_2 \simeq -0.716V_{xr}^* - 0.552V_{zr}^* - 0.416(X_m - X_r^*) \tag{3.10}$$

より，ボールの初速度と飛距離の関係を表している．以上の結果より，入出力マップの入力ベクトルとして $(V_{xr}^*, V_{zr}^*, a_z)$ を，出力として $(X_m - X_r^*)$ を用いることが妥当であると考えられる．

表 3.1 Eigenvalue and Eigenvector of R

	Eigenvalue	Eigenvector
λ_1	2.00	$[\ 0.082,\ 0.439,\ 0.692, -0.567]^T$
λ_2	1.60	$[-0.716, -0.552,\ 0.095, -0.416]^T$
λ_3	0.35	$[-0.688,\ 0.579,\ 0.068,\ 0.432]^T$
λ_4	0.05	$[-0.082,\ 0.408, -0.712, -0.565]^T$

図 3.6 は，取得されたデータ点とともに探索点近傍の 5 点の入出力データを用いて得られた入出力マップを $(V_{xr}^*, V_{zr}^*, X_m - Xr^*)$ の部分空間の中で示したものである．

最小 2 乗法による面近似を行ってデータを補間している．また **図 3.7** は，**図 3.6** のマップを用いて行った打撃位置の推定結果と実際にボールが $Z = 195[mm]$ の高さになったときの X 方向位置の誤差を飛来したボールごとに表示したものである．このときのラケット姿勢は $\theta_4 = -\pi/8[rad]$ に固定している（ラケットをやや下向きに傾けた姿勢）．誤差の平均値は $1.5[mm]$，標本標準偏差は $25[mm]$ であり，予測打撃位置でほぼ確実にボールをラケットでとらえられることがわかる．

3.4.2 ボールの着地点制御結果

前節で述べた打撃位置決定用マップを作成した後に，打撃時のラケットスピードに対応したミラーゲイン決定用マップを作成する．トレーニング時のデータは入力空間内にできるだけ一様に分布させることが望ましいことから，ミラーゲイン k をランダムに与え，ラケットスピードを様々に変化させてボールを打ち返す．ただし，実現できるラケットスピードの上限値 S_{max} を超えるミラーゲインを設定すべきではないため

$$k = \frac{S_{max}}{V_{xr}^*} U_r \tag{3.11}$$

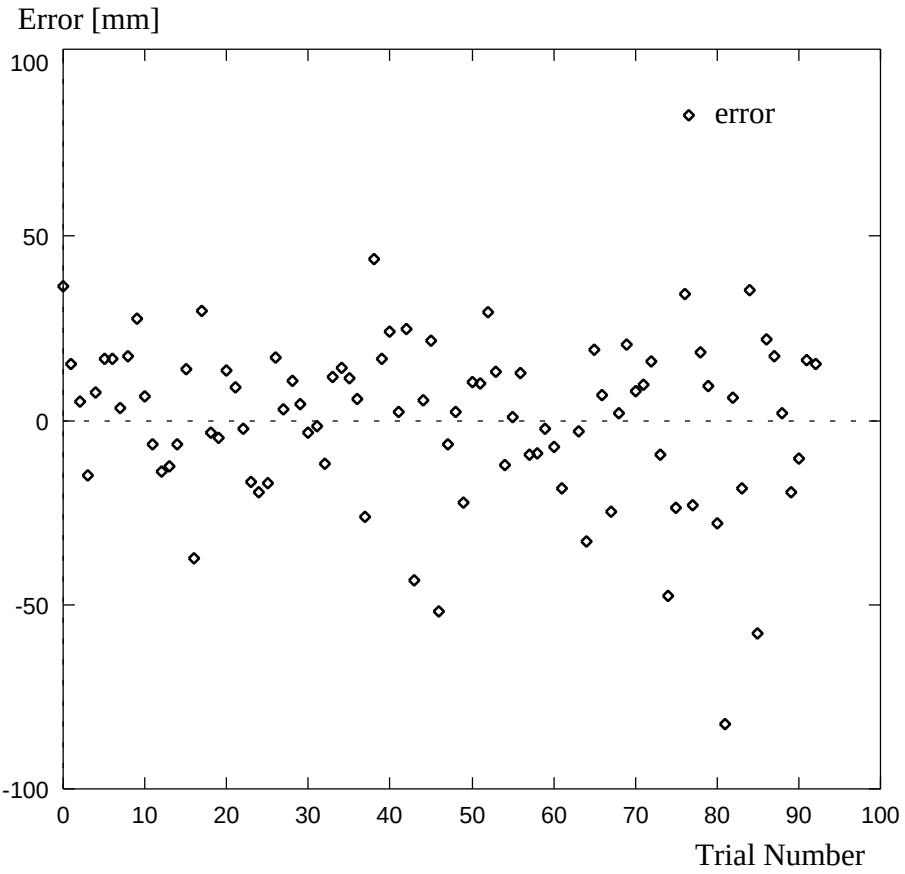


図 3.7　Errors in the Z direction

にしたがってミラーゲインを決定する．ここで U_r は，$0 < U_r < 1$ を満たす一様乱数である．ミラーゲインを設定した後，(3.1) にしたがってラケットを駆動し，打ち返されたボールの着地点 X_g を計測して，入力 $(V^*_{xr}, V^*_{zr}, a_z, X_m - X_g)$，出力 k のペアをマップデータに加えていく．

このようにして作成されたミラーゲイン決定用マップを用いて行ったボールの打ち返し結果の一例を **図 3.8** に示す．◇で示した点は $1/60[s]$ ごとにとらえたボール位置，＋で示した点は $1/30[s]$ ごとのラケット中心位置を表しており，時間の推移を矢印で示している．このときの打撃位置決定用マップで決定された打撃位置は $X_m = 50.2[mm]$，ミラーゲイン決定用マップで決定されたミラーゲインは $k = 0.177$，打ち返したボールの目標着地点は $X_g = -1600[mm]$ であった．飛来するボールをラケット中心（高さ $Z = 195[mm]$）付近でとらえ，目標着地点の近くに打ち返していることがわかる．

このときの速度コマンド，目標動作軌道と実際のラケット軌道および速度をボールの軌跡とともに**図 3.10** に示す．位置，速度ともに指令値より 0.1[s] 弱程度の遅れが見られる．これは，サーボ系システム自体の遅れとロボットの弾性要素に起因すると思われるが，こ

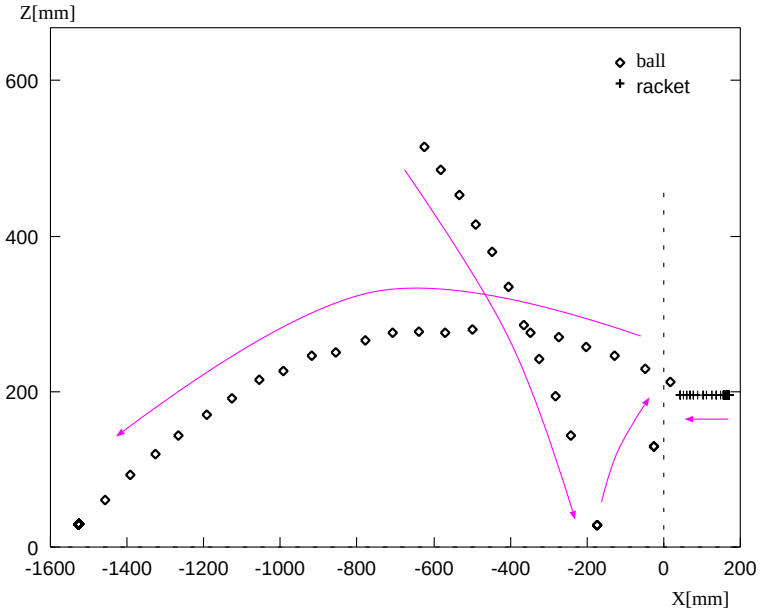


図 3.8 Trajectories of the ball and paddle

の遅れも含めた適当な速度を実現する k の学習が行われているために，打撃が実現できていると考えられる．しかしながら，この遅れのために滑らかな軌道が生成されず，速度が振動して不安定な挙動を示すことが多かった．

また **図 3.9** は，ボールの目標着地点を $X_g = -1700[mm]$ に固定し，飛来するボールの状態を変化させて行った 150 球の打ち返し結果を示している．ボールを打撃するたびにミラーゲイン決定用マップも更新されるので，試行回数が増えるにつれて着地点の誤差が減少していることがわかる．なお，着地点の平均値は $\bar{X}_g = -1694[mm]$，標本標準偏差は $178[mm]$ であり，ほぼ目標着地点付近に打ち返せていた．

3.5 まとめ

本論文では，卓球タスクにおけるストローク動作の主要部分は仮想ターゲットの予測に基づいて実行されるものと考え，その予測のための２つの入出力マップを提案した．その１つである打撃位置決定用マップによってボールの打ち返し位置が決定され，もう１つのミラーゲイン決定用マップによって打撃時のラケットスピードに対応したミラーゲインが決定される．決定されたこれらの仮想ターゲットをミラー則に類似の視覚フィードバック則を用いてラケットの動作に反映させ，ボールをラケットでとらえ目標の着地点付近に打ち返すことができることを示した．ただし，本論文で実現した卓球タスクは必要最小限のラケット自由度と制御パラメータを用いて行ったものであり，打ち返し可能なボールの速度やコースが限定される．また，サーボ系の遅れやシステムの弾性要素により指令値と実際の軌道に遅れが生じていたために，不安定な軌道を生成していた．これらの問題点を解決するため，卓球タスクにおける物理現象を３つに分解したマップにより時刻と位置を予測

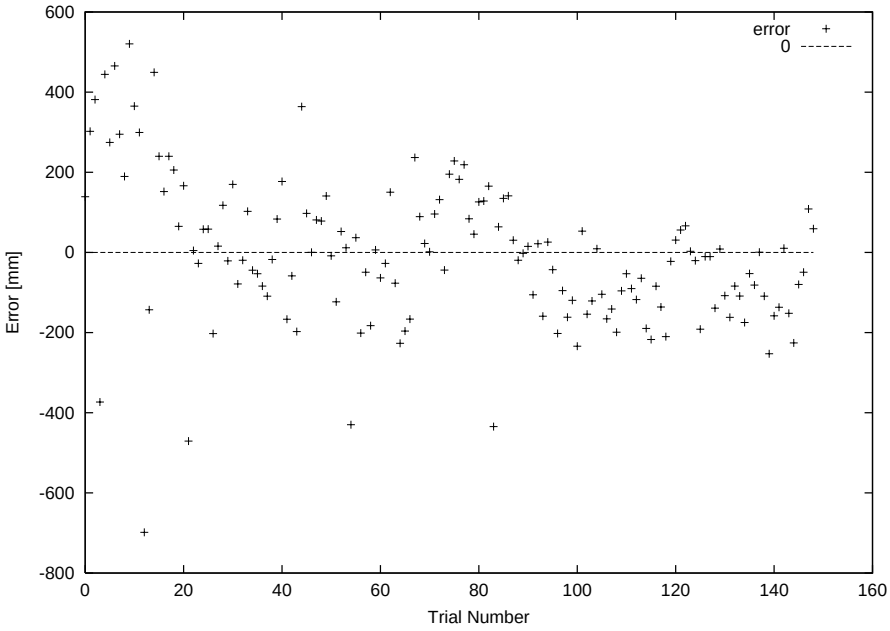


図 3.9　Errors in landing point (X) (target: X=-1700[mm])

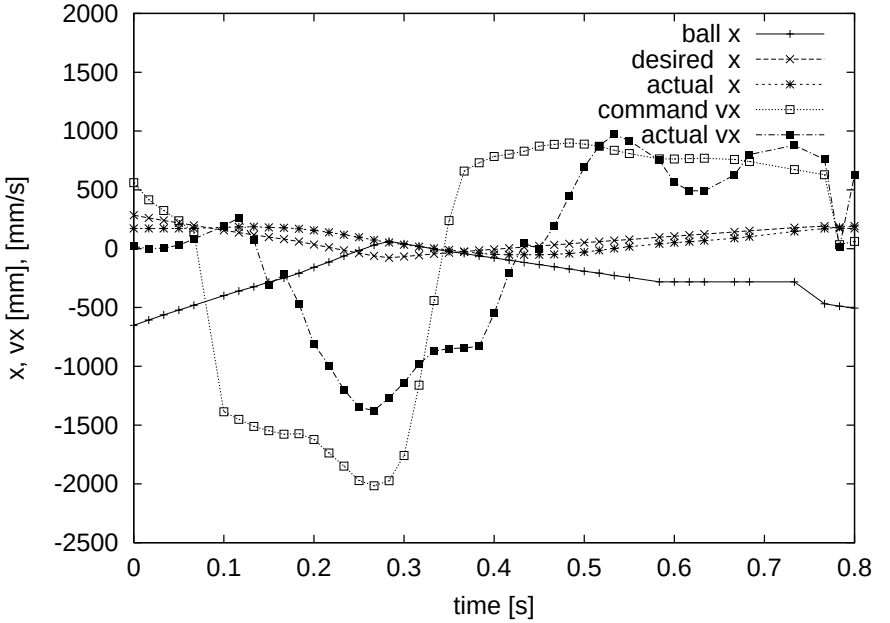


図 3.10　目標軌道，指令速度，実軌道，速度軌道とボール軌道

し，学習制御手法を用いて正確な動作を行うアプローチを採用した．次章で学習制御について述べ，第 5 章にそれを利用したボールコントロールタスクと対人ラリータスクを行った結果について述べる．

第 4章　学習制御

4.1　はじめに

通常実験等で想定している飛来するボールの速度は 5000[mm/s] 前後であり，卓球台が全長 2700[mm] 程度であることを考慮すると，卓球タスクの実現には相手の打撃から 0.5[s] 程度の短い時間で打撃動作を行う必要がある．しかしながら，通常ロボットを短時間で急激に動作させようとすると，サーボ誤差やシステムの弾性の問題から，実際の動作は目標の軌道に対して遅れを生じることになる (**図 3.10**)．このため，サンプリング毎にフィードバックを行うミラー打法では精度の良い動作を行うことができず，不安定な挙動を示した．

卓球タスクを実行するためには，計画した軌道を正確に実現する必要がある．このような高精度な追従制御を可能にするフィードフォワード制御手法のひとつに学習制御 (ILC:Iterative Learning Control) がある [14]．しかしながら，この方法は事前に繰返し学習を行い，目標軌道を実現するような入力コマンド列を求めておく必要がある．ところが，卓球タスクのような動的な環境下でのタスクはボールに合わせた打撃が必要となるため，必要となる打撃軌道のパターンが無数に存在することになる．したがって，必要な打撃パターンを予め学習しておくことは現実的に困難である．

そこで我々は，数通りの多項式目標軌道を実現する制御入力を学習制御によって事前に求めておき，それらを組み合わせることによって任意の多項式目標軌道を実現する制御入力を瞬時に求める方法 (Direct ILC) を提案した．本章では，この Direct ILC 手法について説明し，実際に実機を用いて繰返し学習を行わない未学習の軌道での正確な制御が可能であることを示した．

4.2　Direct ILC を用いたラケット操作

4.2.1　目標軌道の設定

卓球タスクでの打撃動作では動作途中での軌道に制約が無いため，始点及び終点での位置，速度，加速度の条件から目標軌道を定めることになる．また，人間の動作についても一般的には 5 次式で表現できると言われており [24]，本研究でも 5 次の時間関数として位置の目標軌道を与えることにする．このとき，位置，速度，加速度の目標軌道はそれぞれ以下のように表される (4.1)〜(4.3))．

$$x_d(t) = c_1t^5 + c_2t^4 + c_3t^3 + c_4t^2 + c_5t + c_6 \tag{4.1}$$

$$v_d(t) = 5c_1t^4 + 4c_2t^3 + 3c_3t^2 + 2c_4t + c_5 \tag{4.2}$$

$$a_d(t) = 20c_1t^3 + 12c_2t^2 + 6c_3t + 2c_4 \tag{4.3}$$

このとき，初期時刻 $t_0 = 0$ での条件を $x(0) = 0, v(0) = 0, a(0) = 0$ とすると．

$$x(0) = c_6 = 0 \tag{4.4}$$
$$v(0) = c_5 = 0 \tag{4.5}$$
$$a(0) = c_4 = 0 \tag{4.6}$$

となる．また打撃時刻 t_1 での条件を $x(t_1) = X_l, v(t_1) = V_l, a(t_1) = 0$ とすると，

$$x(t_1) = c_1t_1^5 + c_2t_1^4 + c_3t_1^3 + c_4t_1^2 + c_5t + c_6 = X_l \tag{4.7}$$
$$v(t_1) = 5c_1t_1^4 + 4c_2t_1^3 + 3c_3t_1^2 + 2c_4t_1 + c_5 = V_l \tag{4.8}$$
$$a(t_1) = 20c_1t_1^3 + 12c_2t_1^2 + 6c_3t_1 + 2c_4 = 0 \tag{4.9}$$

となる．なお立上り時の加速特性を考慮し，初期時刻および打撃時の加速度を 0 としている．打撃時の加速度を 0 にする点については，人間の計測パターンからも同様の結果を得ており，撃ち返し後のボールを精度良く操作するために必要な条件であると考えられる．

以上より，係数 c_1, c_2, c_3 は (4.10) を解くことによって求まる．

$$\begin{bmatrix} T_l^5 & T_l^4 & T_l^3 \\ 5T_l^4 & 4T_l^3 & 3T_l^2 \\ 20T_l^3 & 12T_l^2 & 6T_l \end{bmatrix} \begin{bmatrix} c_1 \\ c_2 \\ c_3 \end{bmatrix} = \begin{bmatrix} X_l \\ V_l \\ 0 \end{bmatrix} \tag{4.10}$$

4.2.2　学習制御 (ILC) による軌道の学習

ILC:(Iterative Learning Control)

我々の使用するロボットは，トルクが小さく，時間遅れが大きいため，打撃動作中での大きな軌道更新は困難であり，頻繁な軌道更新を行う制御は適さない．また，0.4(sec) 程度の短時間での高速動作を想定しているため，フィードバック方式では，その補償が反映される前に打撃時刻を迎えてしまうことから，我々のロボットには，フィードフォワード方式の制御が適していると思われる．CP(continuous path) 制御を実現するフィードフォワード制御入力を生成する方法として学習制御 (ILC) がある．学習制御では，目標軌道に対する指令軌道を試行毎に修正していき，最終的に目標軌道を実現する指令入力列を獲得する．この学習制御では，制御対象のダイナミクスを厳密に推定することなく，目標軌道を実現するフィードフォワード入力を獲得できる利点があるが，学習に数回から数十回の繰り返し動作を要するため，他の教示方式等と同様に，単一軌道の実現に用いられるのが一般的であり，軌道を変更する度に，繰り返し学習動作を行う必要がある．

(4.2) で定めた速度軌道を目標軌道として，これを実現する制御入力 u を学習的に獲得する方法を示す．まず，目標軌道をサンプリングレート N で離散化した時系列データを $x_d[n]$ と表す．n は $0 \leqq n \leqq T_l/N$ である．同様に時系列の入力，出力をそれぞれ $u[n], x[n]$ で表す，

k 回学習後の入出力をそれぞれ u_k, x_k とし，$k+1$ 回目の入力は，k 回目の結果を元に，以下の (4.11) にしたがって更新される．

$$u_{k+1}[n] = u_k[n] + \Phi(e_k[n]) + \Gamma(\dot{e_k}[n]) \tag{4.11}$$
$$e_k[n] = x_d[n] - x_k[n] \tag{4.12}$$

Φ,Γ は学習オペレータを表し，初回の入力は $u_0 = x_d$ とする，この更新を次の評価関数 $E[k]$ がある閾値より小さくなるか，もしくは $E[k+1] > E[k]$ となるまで繰り返す．

$$E[k] = \frac{\Sigma_{n=0}^{T_l/T} \sqrt{e_k[n]^2}}{\Sigma_{n=0}^{T_l/T} \sqrt{y_d[n]^2}} \tag{4.13}$$

ILC 実験結果

サンプリングレート N は 600[Hz]，学習オペレータは $\Phi = 0.4e^{TS}$,$\Gamma = 0.1^{TS}$ (T は位相進め量で実験では $T = 0.12$[s] とした) として行い，経験から閾値を 0.01 と設定したときの実験結果を示す．実際にはこれ以上評価関数は小さくなるが，その際にはオーバーフィッティングの傾向が見られたため，0.01 で打ち切っている．

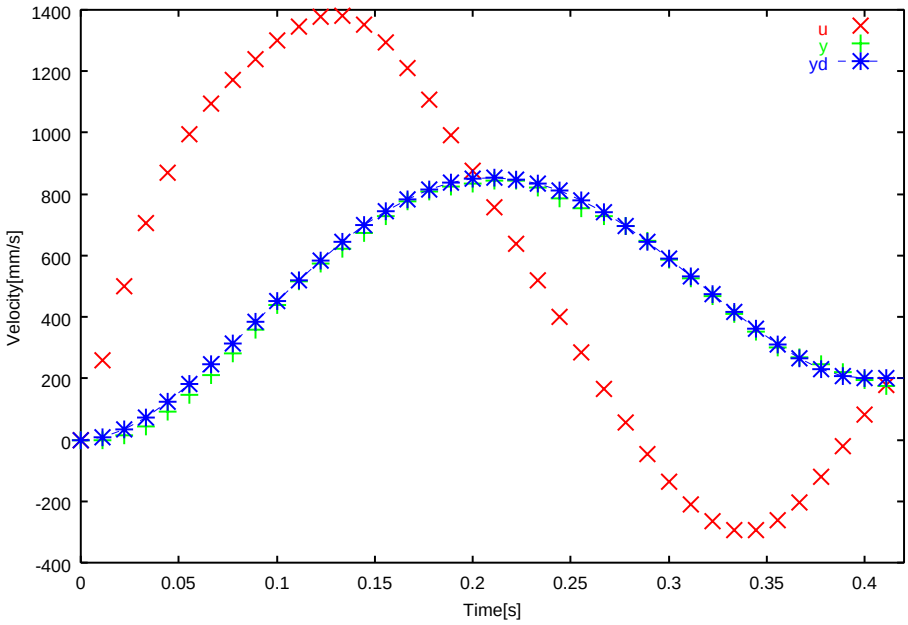


図 4.1 Input and output trajectories after learning:velocity

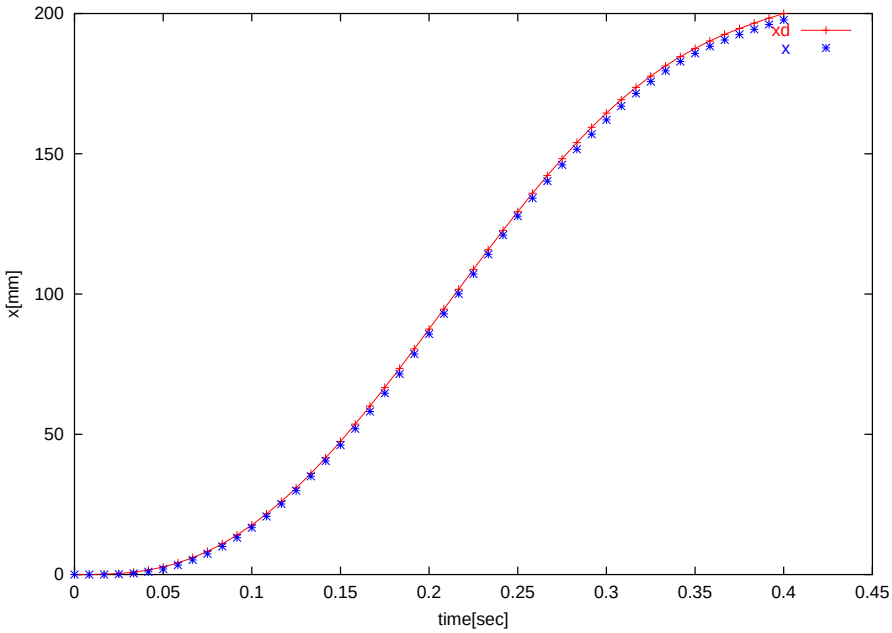


図 4.2　Output trajectories after learning:x

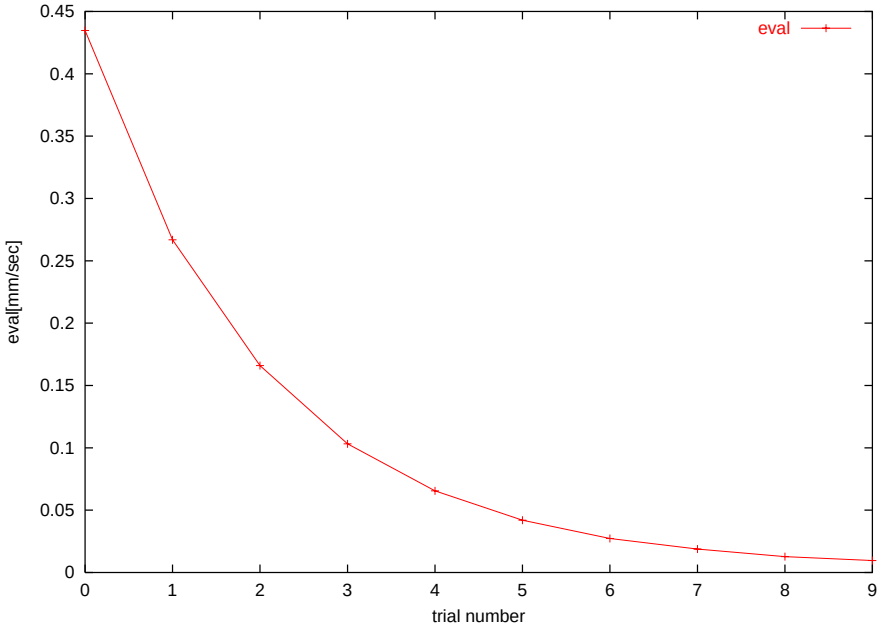


図 4.3　Changes of evaluation

動作時間を $t_1 = 0.4$，目標とする境界条件を，

$$\begin{bmatrix} x(t_1) \\ v(t_1) \\ a(t_1) \end{bmatrix} = \begin{bmatrix} 200 \\ 200 \\ 0 \end{bmatrix} \tag{4.14}$$

として，実験を行った結果を示す．

図 4.1 に学習後の制御入力とそのときの速度，および目標速度軌道，**図 4.2** に学習後の位置の出力と目標軌道を示す．それぞれ，目標軌道に追従する制御入力を生成できていることが分かる．また，**図 4.3** にこのときの評価関数の遷移を示す．9 回目の試行で評価関数 $E[k] < 0.01$ となり，実験を終了している．

4.2.3 未学習の目標軌道への応用

4.1 節で触れたように，卓球タスクにおける打撃の速度パターンは無数に考えられるので，予め全てのパターンについて学習を行っておくことは現実的ではない．そこで，目標軌道が多項式で表されていることとロボットの線形性を利用して，未学習の軌道を正確に実現する入力パターンを繰返し学習を行わずに求めることを考える．

以下にその手順を示す．

(1) 境界条件 $T_L = T_a$,$X_L = X_a$,$V_L = V_a$ を満たす軌道 $y_{d_a}(t) = c_{a1}t^4 + c_{a2}t^3 + c_{a3}t^2$ を出力する時系列入力 u_a を学習制御により学習し，u_a と c_a $[c_{a1}, c_{a2}, c_{a3}]^T$ を一組のデータとして保存する．同様に，$[u_b, c_b], [u_c, c_c]$ の合計 3 組の入出力関係を学習しておく．

(2) 未学習の目標軌道 $y_{d_d}(t) = c_{d1}t^4 + c_{d2}t^3 + c_{d3}t^2$ を，式 (4.15) のように 3 組の学習済み軌道の重ね合わせとして表す．

$$y_{d_d}(t) = k_a y_{d_a}(t) + k_b y_{d_b}(t) + k_c y_{d_c}(t) \tag{4.15}$$

ここで $[k_a,k_b,k_c]^T$ は，

$$\begin{bmatrix} c_{a1} & c_{b1} & c_{c1} \\ c_{a2} & c_{b2} & c_{c2} \\ c_{a3} & c_{b3} & c_{c3} \end{bmatrix} \begin{bmatrix} k_a \\ k_b \\ k_c \end{bmatrix} = \begin{bmatrix} c_{d1} \\ c_{d2} \\ c_{d3} \end{bmatrix} \tag{4.16}$$

から求まり，y_{d_d} を出力する入力 u_d を学習済みの 3 パターンの入力 u_a, u_b, u_c の重ね合わせ

$$u_d = k_a u_a + k_b u_b + k_c u_c \tag{4.17}$$

として得る．

4.2.4　実機を用いた Direct ILC の検証実験

Direct ILC の有効性を示すために，**表 4.1** に示した境界条件 (3 パターン) の学習済み入出力データを用いて未学習の入力を生成する検証実験を行った．それぞれの目標軌道と通常の ILC によって得られた入力列を**図 4.4** に示す．目標軌道を，(4.14) の境界条件で表される動作パターンとして与えたときの係数は，$k = [-2.058824, 3.906250, -3.573529]^T$ である．

表 4.1　用いたパターンの境界条件

パターン	動作時間 [s]	終端位置 [mm]	終端速度 [mm/s]
A	0.4	20	800
B	0.5	200	0
C	0.6	150	800

Direct ILC によって繰返し学習を行わずに合成された入力列は，通常の ILC によって繰返し学習を行って獲得された制御入力と十分一致する (**図 4.5**)．以上より，学習済みの軌道の重ね合わせによって新たな未学習の軌道を実現する適切な入力列を即座に作成できることが示された．そこで，この Direct ILC 手法を卓球のストロークに応用する．

4.3　卓球用動作軌道

卓球のストロークは，大きく分けて 2 つの部分からなっていると考えられる．静止状態から打撃を行う “打撃動作” と打撃後に次の打撃に備えるために待機位置へ戻る “帰還動作” である．打撃を行うための適切な仮想ターゲットが与えられると，その仮想ターゲットに対応する最終位置 (p)，速度 (v)，加速度 (a) をもつラケットの打撃軌道を決定することができる．各軸に対し 5 次の多項式による位置軌道としてその打撃軌道を表し，したがって，速度軌道を以下の形で設定した．

$$v(t) = c_1 t^4 + c_2 t^3 + c_3 t^2 \tag{4.18}$$

また，境界条件を以下のように設定した：

$$\begin{array}{lll} \mathrm{p}(0) = p_i, & \mathrm{v}(0) = 0, & \mathrm{a}(0) = 0 \\ \mathrm{p}(t_h) = p_h, & \mathrm{v}(t_h) = v_h, & \mathrm{a}(t_h) = 0 \end{array} \tag{4.19}$$

ここで t_h はロボットのトルク制限を考慮した軌道全体の動作時間である．これらの条件からラケット動作は静止状態から開始され，加速度 0 で終了することが分かる．係数 c_1, c_2, c_3 はこれらの境界条件を適用することで求まる．ラケットがボールの打撃を実現した後，ラケットはできる限り速やかに待機位置へと帰還しなくてはならない．式 (4.20) の境界条件を反転して同じ式 (4.18) を用いることで帰還動作を作成した．

ただし，ILC の軌道生成において，初期位置，時刻，速度を 0 にするという条件があるので，システムの線形性を前提とし軌道を平行移動してもコマンドと実際の動作軌道の関係が維持されると考え，以下の条件に置き換えて “帰還動作” のコマンド生成を行った．

$$\begin{array}{lll} \mathrm{p}(0) = 0, & \mathrm{v}(0) = 0, & \mathrm{a}(0) = 0 \\ \mathrm{p}(t_h) = \text{-}p_h, & \mathrm{v}(t_h) = \text{-}v_h, & \mathrm{a}(t_h) = 0 \end{array} \tag{4.20}$$

図 4.6 は，$p_h = 500[\mathrm{mm}], \mathrm{v_h} = 200[\mathrm{mm/s}], \mathrm{t_h} = 0.4[\mathrm{s}]$ の条件で 1 ストロークの目標軌道を設定し，実際に Direct ILC によりコマンド列を生成して実際に動作させた結果である．これにより，提案手法によって卓球のストロークが正確に実現できることが示された．

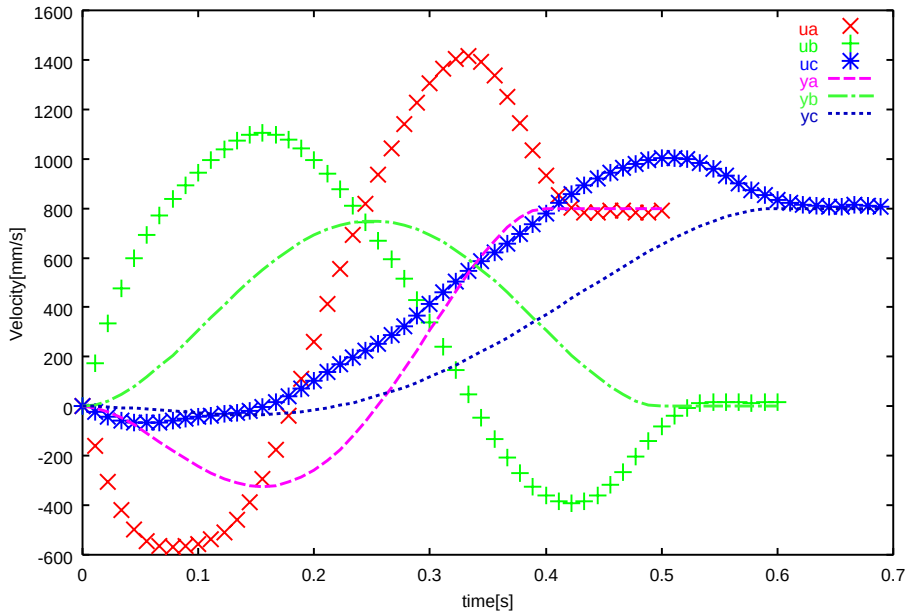


図 4.4 Patterns A, B and C (u*i*: Learned input. y*i*: Desired output.)

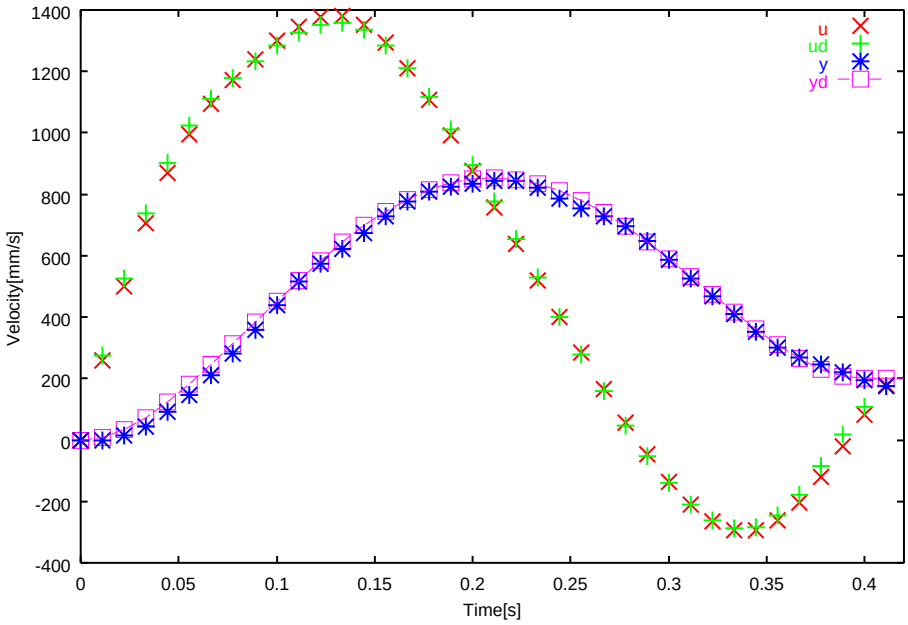


図 4.5 A new input and output trajectory generated by patterns A, B, and C (u_d: Input calculated by the proposed method, u_{ILC}: Input obtained by ILC, v: Actual trajectory, v_d: Desired trajectory)

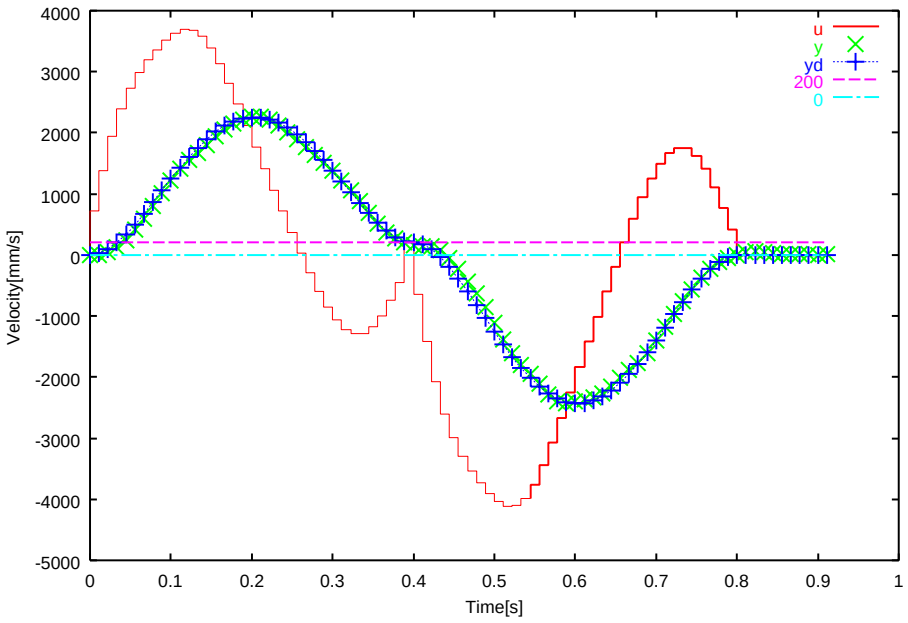


図 4.6 One stroke movement (u_d: Input calculated by the proposed method, v: Actual trajectory, v_d: Desired trajectory)

第 5 章　ボールコントロールタスク

5.1　はじめに

卓球タスクは，簡単に言えば飛来するボールを相手コートに打ち返すことである．しかしながら，その質を高めようとするならば，ただ打ち返すだけでなく，狙った位置へ狙った軌道で打ち返すことが重要である．そのためには，ロボットはパドルの速度や位置，姿勢を適切なタイミングで調整し，打撃動作を行わなければならない．

本章では，局所重み付き回帰（LWR）を利用した次の 3 つの入出力マップを用いて打撃動作計画を行い，飛来するボールを任意の飛行時間後に望みの落下位置へと打ち返すボールコントロールタスクについて述べる．

(1) 飛来するボールの状態を記述する入力ベクトルに対し，打撃時刻とその時のボール位置および速度を予測するマップ,

(2) 打撃前後のボール速度の変化を示すマップ,

(3) 打撃直後のボールの速度と打ち返したボールの着地点および飛行時間の関係を示すマップ.

5.2　1 ストロークとボールイベント

以降の説明の便宜上，**図 5.1** および以下に示されたボール状態の変化に注目し,「ボールイベント」を定義する.

5.2.1　ボールイベントの定義

Event- (s) 相手選手による打撃

Event- (m) 計測仮想平面通過

Event- (l) ロボット側コートでの反発

Event- (h) ロボットによる打撃 (または打撃仮想平面通過)

Event- (r) 相手側コートでの反発

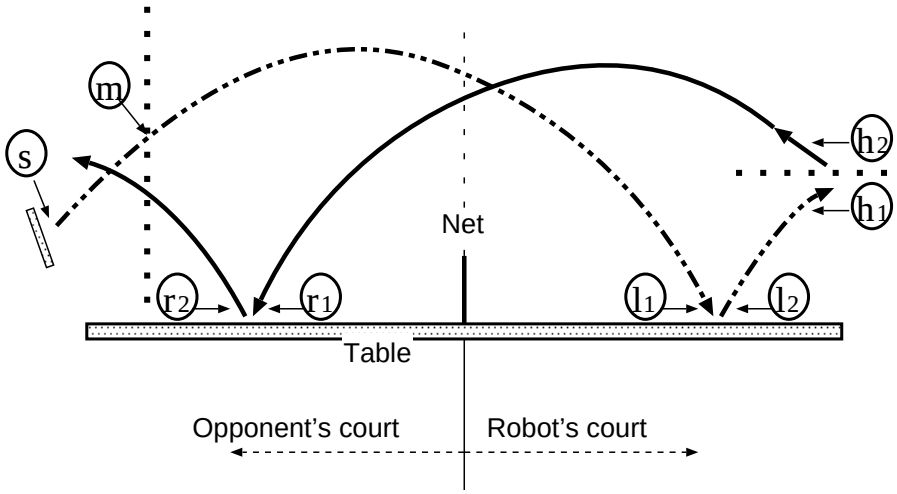


図 5.1　Definitions of ball events

図 5.1 における添字の 1,2 はそれぞれ反発の “直前” と “直後” を表す．相手もしくは打球機がボールを打撃 (s) し，飛来するボールの挙動を計測するために設定された仮想平面 (VP:virtual plane) を通過 (m) した時点で，このタスクは開始される．(仮想平面の位置は **図 5.8** に示すように，$x = -1000$[mm] である)．

ボールはネットを越えた後，ロボット側のコートでバウンド (l_1, l_2) し，ロボットによって打撃される (h_1, h_2)．その後，打ち返されたボールは再び相手側のコートでバウンドし (r_1, r_2)，次に相手によって打撃されるか，ボールがコートで 2 度目のバウンドをするか，もしくは見えなくなるまでが 1 回の打撃となる．

5.2.2　打撃動作

卓球タスクを 3.2.1 節と同様に，3 つのサブタスクに分割する．以下にそれぞれのタスクについて要約し，**図 5.2** に図示した．

TASK A: 予測と打撃軌道の計画を行うタスク

TASK B: TASKA で計画した打撃および帰還動作を実現する打撃帰還タスク

TASK C: 入出力マップ (5.3 節で説明) を更新し，引き続きボールの動きを監視してイベント m の発生を待つ待機タスク．

さらにタスク A を 2 つに分ける．A_1 では，システムはボールを打つために必要なパラメータを予測し，仮想ターゲット (打撃時刻，ラケット姿勢および打撃速度，5.3 節参照) を決定する．A_2 では，A_1 で行った予測をもとに，打撃軌道とその軌道を実現する動作コマンドを DirectILC 手法により生成する (4.2 節参照)．

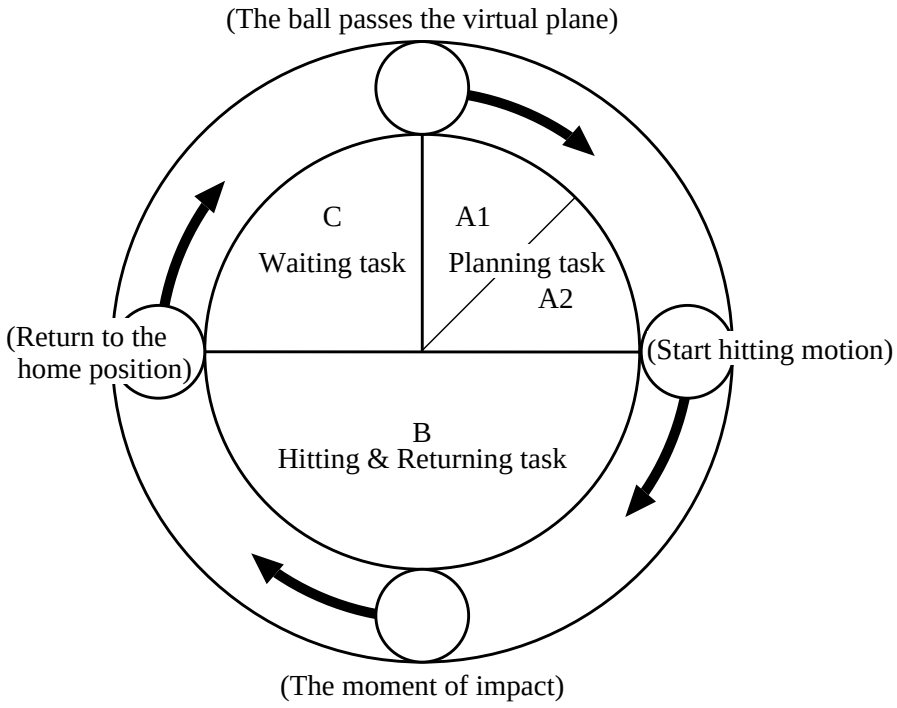


図 5.2 One stroke movement

5.2.3 各イベントにおけるボール状態の推測

このアプローチでは，ロボットの動作はイベント m でのボールの状態 (位置，速度，加速度) に基づいて計画される．この時のボール状態の推定手順を次に示す：

① ボールが仮想平面を通過するまで，QuickMAG によって得られたボールの 3 次元位置を記憶する．

② ボールの軌道を X と Y それぞれの方向を時間の 1 次式として，また Z 方向を時間を 2 次式として，最小二乗法を用いて近似する．

③ 2) で得られた軌道を用いて，イベント m でのボール状態を求める．

同じ手順により，他のイベントでのボール状態も推測する．

5.3 入出力マップによるラケット動作の決定

卓球タスクのようなスポーツに限らず，人間が生活する中で多くの予測という行為を行う．『とにかくボールにラケットを当てる』という卓球において最も基本となるタスクを行なうだけでも，少なくとも打撃位置を予測する必要がある．第 3 章では，ボール軌道の打

撃位置のみを予測し，打撃動作はミラー則に従って生成することでボールの打撃位置での捕捉を可能としたが，本章で提案した打撃動作方法を用いるためには，打撃時刻も予測する必要がある．また，打撃の瞬間のボール状態をできるだけ詳しく予測した方が，打撃後のボール軌道を操る上で有利になると考えられるため,「打撃時刻，及び打撃時のボール状態の予測」は卓球タスクにおいて非常に重要である．更に，これら予測は反発が生じる前，しかもロボットがその予測結果を実現するために必要な時間を残して完了されなくてはならない．

ボールの空力学とテーブル上でのバウンドを表す物理学はイベント m と h_1 の間でのボールの状態変化を支配する．Anderssson はボールの運動方程式を次のように公式化した：

$$\boldsymbol{a} = -C_d|\boldsymbol{V}|\boldsymbol{V} + C_m|\boldsymbol{V}|\boldsymbol{W} \times \boldsymbol{V} - g \tag{5.1}$$

ここで，$\boldsymbol{V}$ は速度ベクトルであり，$\boldsymbol{W}$ はスピンベクトルである．$\boldsymbol{a}$ は加速度ベクトルであり，g は重力加速度である．C_d と C_m はそれぞれ摩擦係数とマグヌス効果の係数である．また，Andersson はボールのバウンドの代数方程式も導いたが，本研究では，ボールの打撃時運動予測にボールの飛行ダイナミクスやテーブル上での反発特性を表現する陽なモデルを用いない．なぜなら，モデルパラメータの同定が繁雑であり，かつモデルが固定されることで，経験と共にロボットのタスク遂行能力が向上する余地が無いためである．

一方，(5.1) から打撃時のボール状態が過去のボール状態から決定されることが分かる．つまり，ボールが相手コート内に設けた仮想的な平面 (VP:Virtual Plane) を通過する際の運動状態を入力，打撃時のボール状態を出力とする入出力マップを作成して打撃時のボール状態の予測を行うことができる．このような memory-based の手法を用いることで，経験の増加と共にタスクの遂行能力を向上させることができる．

これらの入出力マップは**図 5.3** ～ **図 5.5** に示した 3 つの物理現象を表す．

マップ 1 – イベント m での飛来ボールの状態を表す入力ベクトルを用いてボールが打撃される時刻とその時 (イベント h_1) のボール位置と速度を予測するマップ．

マップ 2 – 速度 (V_h) と姿勢 (θ_3, θ_4) で打撃されたときのボールの打撃前後 (h_1, h_2) の速度変化を示すマップ．

マップ 3 – 打撃直後のボール速度と打ち返されたボールのバウンド位置 (p_r) および時間 (t_r) との関係を表すマップ．

これらのマップは LWR(APPENDIX 参照) によって実装する．記憶しているデータに対して，LWR を用いて予測点における入出力関係を二次元局所モデルに近似する．回帰において，全てのデータ点はそれぞれの理想点との距離の関数で重みを与えられ，モデルのパラメータは最小二乗法で決定する．

マップ 1 の学習は，人間あるいは打球機によって打ち出されたボールを計測し，イベント m と h_1 でのボール状態の対応関係を記憶することによって行う．一旦マップ 1 が作成されれば，ロボットはそのマップを用いて様々なボールに対し打撃位置と時刻を予測できるので，“打撃” を実現できる．そこで，ラケットの姿勢と速度をランダムに選んで打撃を行

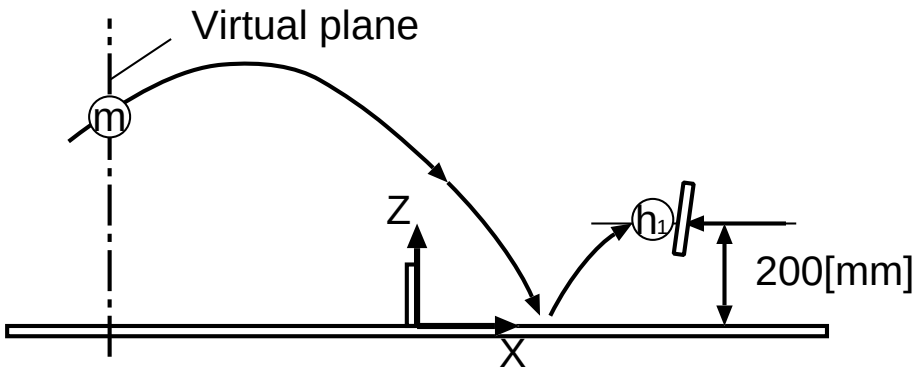


図 5.3 [マップ 1] – 飛来するボールの状態 (m) を表す入力ベクトルからボールの打撃時刻と位置およびそのとき (h_1) の速度を予測するマップ.

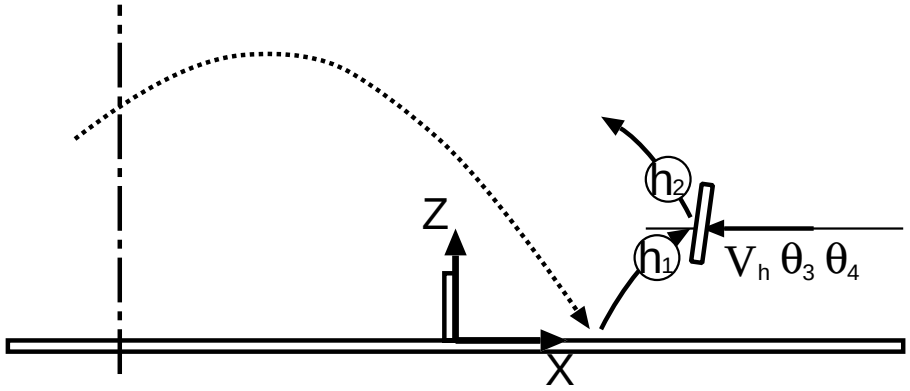


図 5.4 [マップ 2] – ラケット状態 $([V_h, \theta_3, \theta_4])$ と打撃前後のボール速度の変化 $(h_1 \rightarrow h_2)$ の関係を表すマップ.

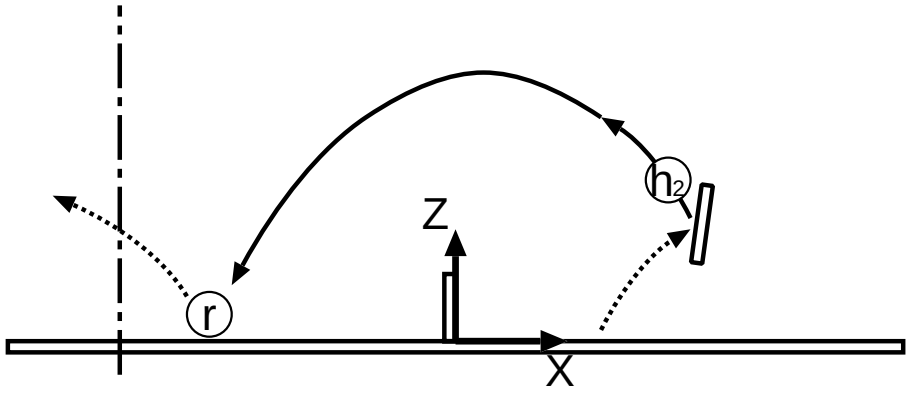


図 5.5 [マップ 3] – 打撃後 (h_2) のボールの速度と打ち返されたボールの跳ねる位置 (r) および時間の関係を表すマップ.

い，そのときのラケットの姿勢/速度と，ボールを計測して打撃前後のボールの速度，および打ち返されたボールの跳ねる位置/時刻の組を記憶することによって，マップ 2，マップ 3 を作成する．これら 3 つのマップについて，それぞれ詳しく説明する．

5.3.1　マップ 1 による予測

予測には多くの種類が存在するが，ここで扱うボールの軌道予測は受動的予測 [25] に相当し，相手選手によって打撃された飛来するボールの未来の軌道を変更することはできない．また，本論文で使用する卓球ロボットは，高さが卓球台から 200[mm] の平面上をラケット中心が移動する構造となっているため，ボールをラケット中心で捕らえる場合，打撃位置は，ボールがネットを越え自コートでバウンド後，高さ 200[mm] となる位置か，頂点通過後に再び落下して高さ 200[mm] となる位置の 2 通りである．ショート打法ではバウンド直後 (上昇中) を打撃する方が，打球をコントロールしやすい [26] ことを考慮し，前者を打撃位置とした．このとき，打撃時刻及び打撃位置は各ボール軌道に対して一意に決定される完全に受動的な予測となる．

5.3.2　マップ 1 の入出力の定義

打撃時 (イベント h_1) でのボール状態はイベント m でのボール状態，つまり，ボールの位置ベクトル $(p_{bmx}, p_{bmy}, p_{bmz})$，速度ベクトル $(v_{bmx}, v_{bmy}, v_{bmz})$ とスピンベクトル $(w_{bmx}, w_{bmy}, w_{bmz})$ に依存していることに注目する．しかし，直接スピンベクトルを観測することは不可能である．(5.1) はどの時点でも成立するので，我々はスピンベクトルの代わりに，イベント m でのボール状態の一成分として，加速度ベクトルを導いた．さらに，ほとんどのスピンは Y 軸まわり (w_{bmy}) であることを考慮して，我々は w_{bmy} の代わりに a_{bmz} を用いる．さらに，イベント h_1 が発生する条件 $p_{bhz} = 200[\mathrm{mm}]$ を考慮すると，イベント m と h_1 の間でのボール状態の変化は，次の形の非線形入出力関係として表現できる：

$$[p_{bmz}, v_{bmx}, v_{bmy}, v_{bmz}, a_{bmz}] \rightarrow [dt, dx, dy], \boldsymbol{V}_{bh1} \tag{5.2}$$

ここで，$dt = t_h - t_m$, $dx = p_{bhx} - p_{bmx}$, $dy = p_{bhy} - p_{bmy}$ であり，$\boldsymbol{V}_{bh1} = [v_{bh1x}, v_{bh1y}, v_{bh1z}]$ である (**図 5.8** 参照)．

マップ 1 の学習

マップ 1 の学習階段では，イベント m と h_1 のボール状態を入出力の組として記憶する．マップ 1 の構築は以下の手順で行う：

① 打撃を行わず，飛来するボールの軌道を計測する．

② イベント m におけるボール状態 $(\boldsymbol{B_m})$ を仮想平面付近におけるボールの位置データから求める．

③ イベント h におけるボール状態 ($\boldsymbol{B_h}$) を打撃平面付近のおけるボールの位置データから求める．

④ 各入力ベクトルを独立に，出力の分散が 1 になるように標準化する．

⑤ Cross validation error check[11] を用いて信頼できるデータのみを格納する．

①は，2.2.1 節で示した QuickMAG を用いて，ボールの 3 次元位置を 60[Hz] のサンプリングレートで計測する．したがって，②，③のボールの運動状態は計測した離散的な位置データから推定する必要がある．

まず，VP の通過判定について説明する．入力値の計測に用いる VP は，y-z 平面に平行に設定される (**図 5.8**)．通過の判定は，**図 5.6** の様に現在の計測点と 1/60[s] 前の計測点を用いて $x_{ball_{old}} < VP_x < x_{ball_{now}}$ から行う．出力値の計測に用いる VP(x-y 平面に平行) に対する通過判定も同様に，位置データを用いて $z_{ball_{old}} < VP_z < z_{ball_{now}}$ から行う (**図 5.7**)．

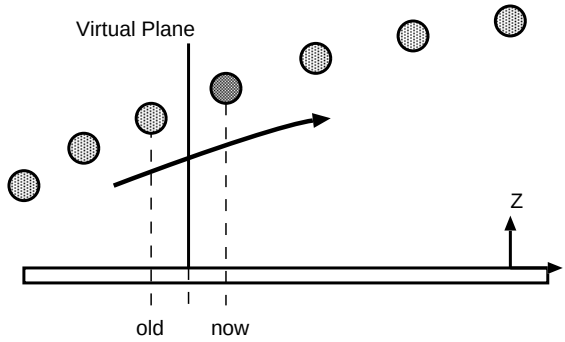


図 5.6 VP の通過:x

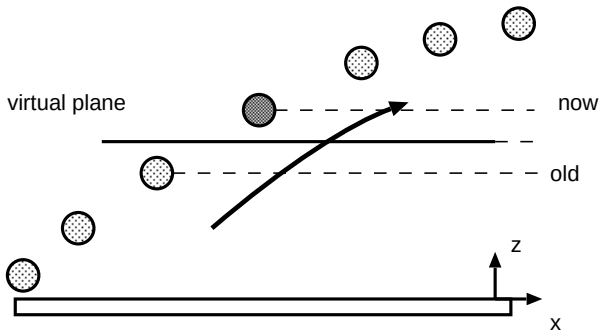


図 5.7 VP の通過:z

計測されたボールの軌道を以下の様に近似する．

$$x = C_{x1}t + C_{x0} \tag{5.3}$$
$$y = C_{y1}t + C_{y0} \tag{5.4}$$
$$z = C_{z2}t^2 + C_{z1}t + C_{z0} \tag{5.5}$$

x 軸，y 軸方向の運動は，それぞれ時間の 1 次式，z 軸方向の運動は時間の 2 次式で近似を行い，計測した位置データをローカルに 7 点用い，最小二乗法を用いて係数を求める．

したがって，ボールの速度や加速度は次のように推定される．

$$v_x = C_{x1} \tag{5.6}$$
$$v_y = C_{y1} \tag{5.7}$$
$$v_z = 2C_{z2}t + C_{z1} \tag{5.8}$$
$$a_z = 2C_{z2} \tag{5.9}$$

マップ 1 の利用

マップの利用段階では，宮崎ら [23] と同様に Locally Weighted Regression(LWR) を用いて学習的に予測を行う (付録 A.2 参照). 彼らと異なるのは，ボールの各軸方向運動の干渉を意識した設計とした点である．これは，当時よりもコンピュータの処理能力が向上したため可能となった．宮崎ら [23] の場合は，マップの入出力数を増やすことにより，処理時間が増加し予測完了後の動作時間が確保できないこと，学習効率が悪くなることからマップの入出力数を最小限に減らす工夫を施していたが，その論文中でも記されている通り，様々なボールへの対応には入出力数の増加は避けられないと考えられる．

打撃時刻 ${}^*t_h$ と打撃位置 $({}^*p_{bhx}, {}^*p_{bhy})$，そのときのボール速度 $({}^*v_{bh1x}, {}^*v_{bh1y}, {}^*v_{bh1z})$ が LWR と以下の関係を用い，マップ 1 によって予測される：

$$
\begin{aligned}
{}^*t_h &= t_m + {}^*dt && (5.10)\\
{}^*p_{bhx} &= p_{bmx} + {}^*dx && (5.11)\\
{}^*p_{bhy} &= p_{bmy} + {}^*dy && (5.12)
\end{aligned}
$$

ここで，${}^*dt, {}^*dx,$ および ${}^*dy$ は補間されたマップ 1 の出力である．

ボール速度 $({}^*v_{bh1x}, {}^*v_{bh1y}, {}^*v_{bh1z})$ も LWR 補間を用いてマップ 1 により予測される．

マップ 1 による予測評価実験

本手法の予測精度を評価するための実験を行った．この評価は，打球機の設定をナックル (無回転) に設定し，そのボール打ち出し位置，角度，速度を変更することによって得られたボール軌道データを用いた．

評価は，あらかじめ取得した軌道データからオフラインでボール状態データを抽出し，データベース用 200 球と予測精度評価用に分割したバリデーションテストにより行う．以下は，それらを用いて予測を行いその精度を評価した結果である．p_{ix} は-1000[mm] とし，その位置でのボール状態 $\boldsymbol{B}_{bi}$ の推定には x=-1400∼-1000[mm] の時系列位置データを用いた．

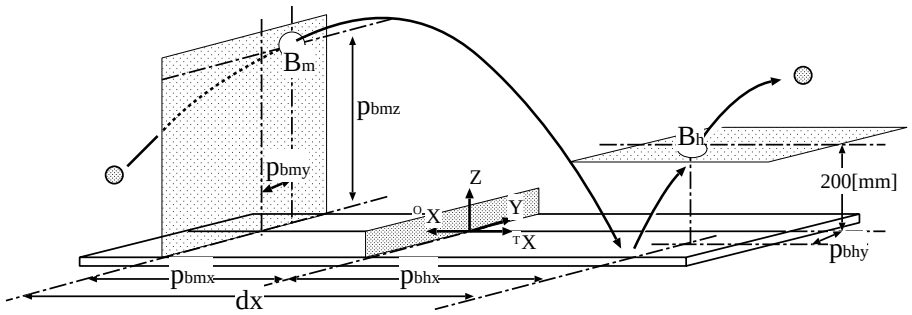


図 5.8　ボール軌道の予測

図 5.9 に打撃時刻予測とその誤差を，**図 5.10** に打撃位置予測とその誤差を，また，**図 5.11** には打撃時ボール速度予測とその誤差を示す．ここでは，あらかじめデータベース内のデータの統計値を計算し，各入力次元独立に標準化を行っている．また，LWR の距離関数の重み行列は単位行列 I，重み関数の band width h は 0.8 一定としている．

これらの図から，ナックルボールの打撃時刻，打撃位置，打撃時の速度を，およそ 0.5[s] 前に予測可能であることがわかる．打撃時刻予測の誤差は msec，打撃位置予測の誤差は cm オーダーであり，この程度の誤差であれば，予測時刻に予測位置へとラケットを運べば，ほぼラケット中心でボールを捕らえることが可能である．また，打撃時速度の予測誤差は cm/sec オーダーであり，この予測速度をそのまま打撃時のラケット状態計画の作成に用いることが可能であると言える．

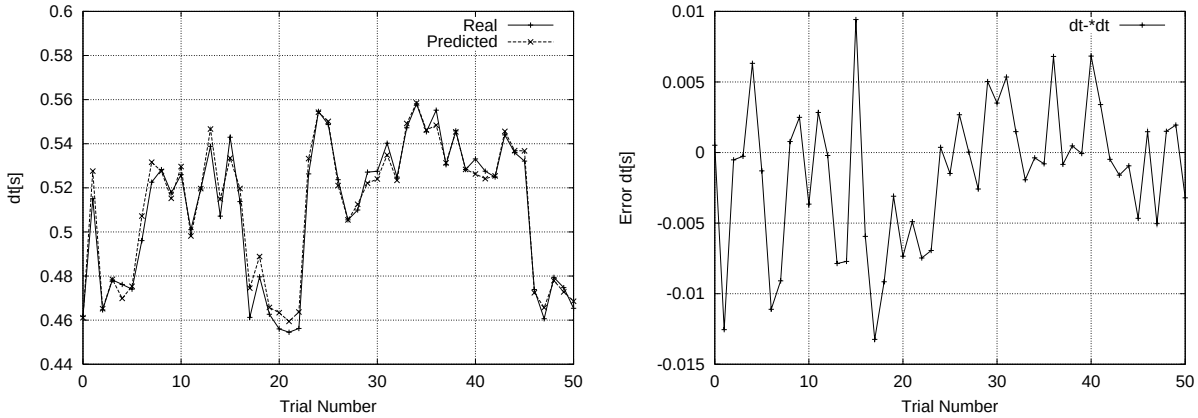


図 5.9　打撃時刻予測とその誤差

5.3.3　マップ 2 およびマップ 3 を用いたラケットの打撃状態の決定

次に，マップ 1 による打撃位置の予測を基に適当な打撃速度および角度で打撃を行い，打撃時のラケット速度とボールの挙動の関係を示すデータを蓄積する．このデータはマップ 2 およびマップ 3 を構築する．仮想ターゲットの残りのパラメータ (打撃姿勢，打撃速度) はマップ 2 とマップ 3 を用いて決定する．

Map2 と Map3 の学習

ボールとラケットの衝突の力学はイベント h_1 からイベント h_2 へのボールの状態変化を支配している．打撃後の飛行の力学はイベント h_2 からイベント r へのボールの状態変化を支配している．これらの変化は以下の形の入出力マップで表されると考えられる．

$$[V_h, \theta_3, \theta_4] \rightarrow \boldsymbol{V}_{bh12} (= [v_{bh12x}, v_{bh12y}, v_{bh12z}]) \tag{5.13}$$

$$\boldsymbol{V}_{bh2} \rightarrow [dt_{hr}, d\boldsymbol{p}_{bhr} (= [dp_{bhrx}, dp_{bhry}])] \tag{5.14}$$

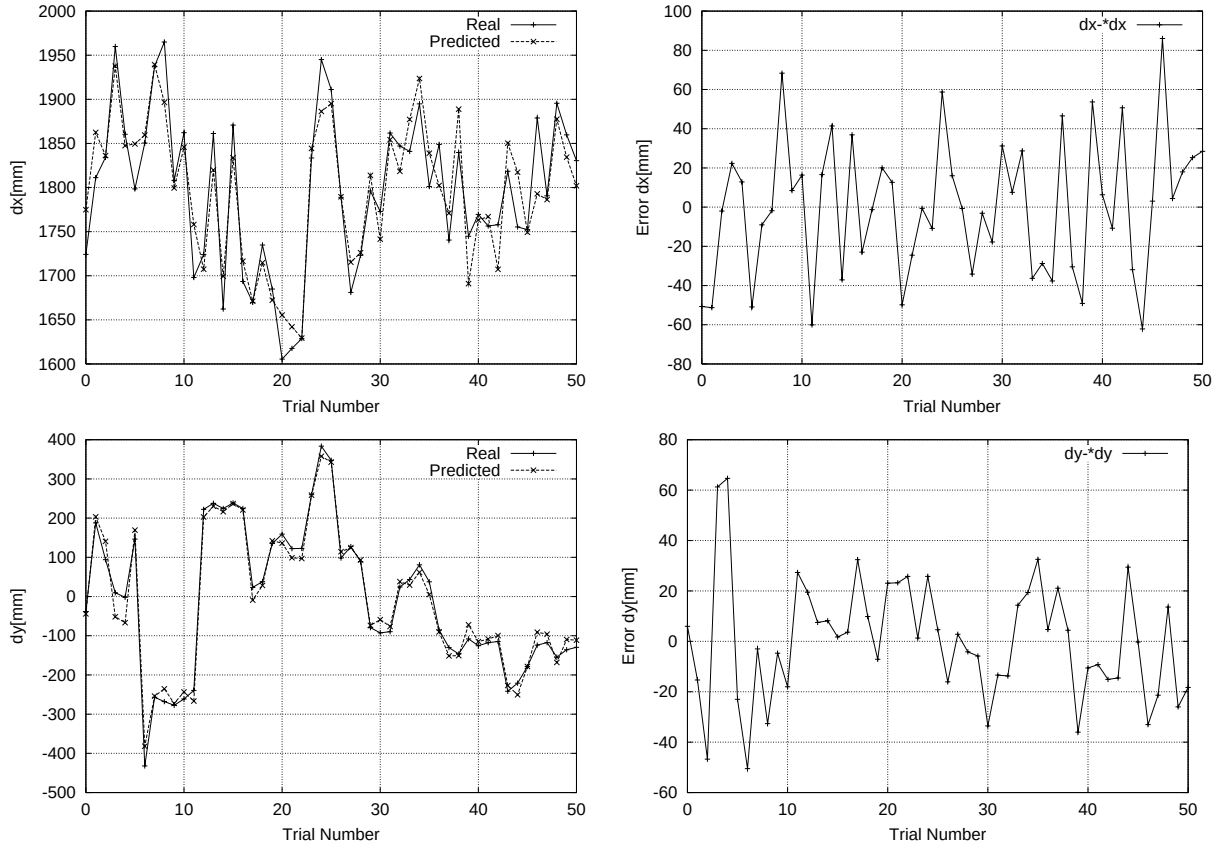


図 5.10　打撃位置予測及び誤差

ここで，$\boldsymbol{V}_{bh12}$ は打撃直前直後のボール速度の変化 ($\boldsymbol{V}_{bh1}, \boldsymbol{V}_{bh2}$) を表している．つまり，$\boldsymbol{V}_{bh12} = \boldsymbol{V}_{bh2} - \boldsymbol{V}_{bh1}$ である．dt_{hr} は返球したボールの飛行時間 (打撃から相手コートのバウンドまで) であり，$d\boldsymbol{p}_{bhr}$ は飛行距離，つまり，$d\boldsymbol{p}_{bhr} = \boldsymbol{p}_{br} - \boldsymbol{p}_{bh}$ である．学習段階で取得されたデータは入力空間に一様に分布しているべきであるので，入力ベクトル $(V_h, \theta_3, \theta_4)$ はランダムに選択した．これらのラケット条件をマップ 1 を用いて決定した打撃位置において実現することによりボールの打撃結果の計測を行った．

マップ 2，マップ 3 の利用

返球を含む (5.13) と (5.14) のマップを順マップとするならば，適切な制御変数 V_h および (θ_3, θ_4) を得るためにはそれらの逆マップが必要である．Andersson はボールとラケットの衝突のダイナミクスをラケットの法線が打撃速度と平行で，かつボールにスピンの無い特別な場合としてモデル化した [5]．そのモデルによると，ラケットの姿勢 (θ_3, θ_4) はボールの速度変化 V_{bh12} によって唯一に決まり，ラケット速度 $V_h(=[v_{hx}, 0, 0])$ は V_{bh12} と V_{bh1}·V_{bh12} によって唯一に決まる．(“·” は内積を示す)．彼のモデルを参考にし，我々は次の逆マップが制御変数を決定すると考えた．

$$[dt_{hr}, d\boldsymbol{p}_{bhr}] \quad \rightarrow \quad \boldsymbol{V}_{bh2} \tag{5.15}$$

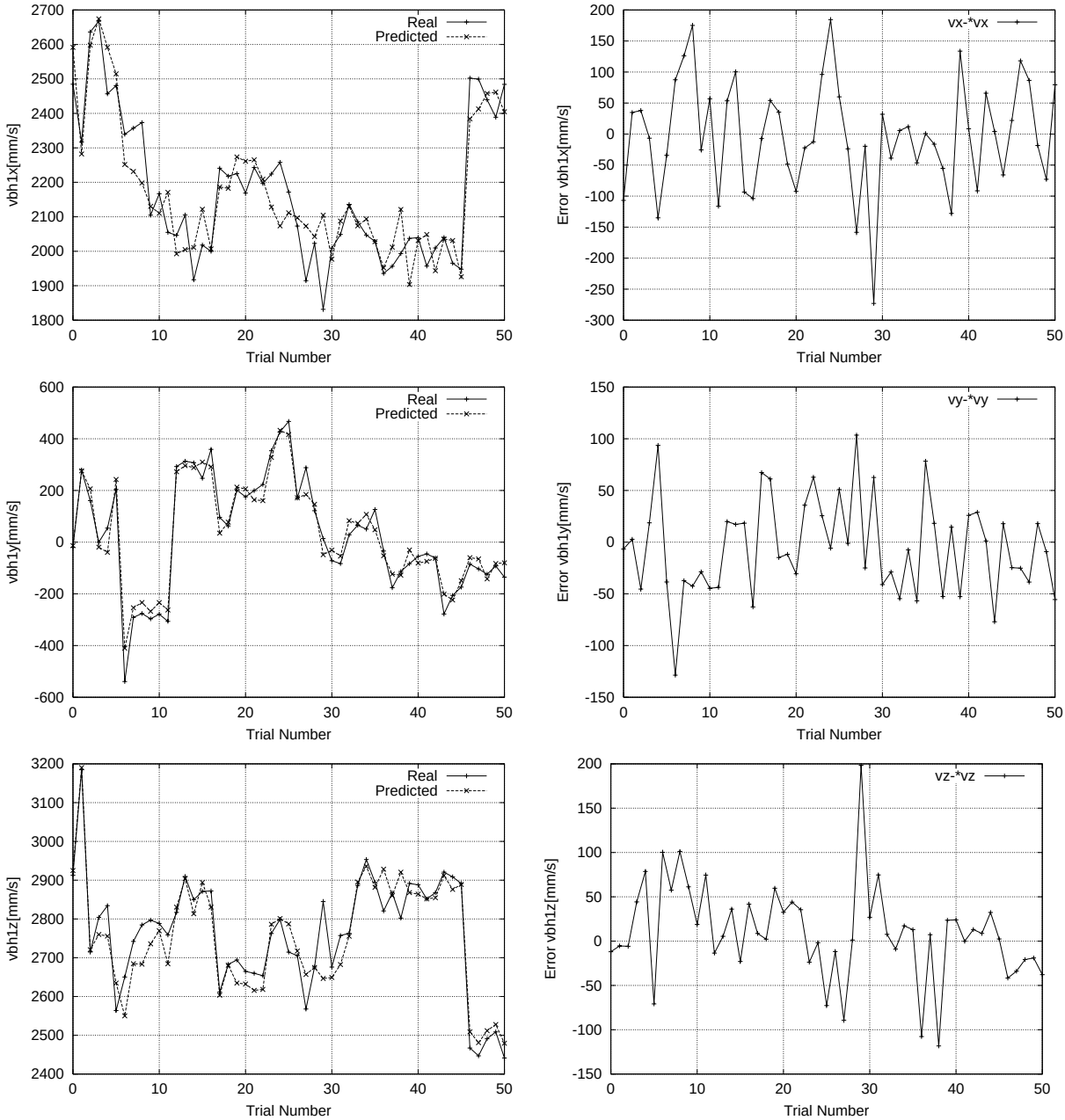


図 5.11 打撃時ボール速度予測及び誤差

$$[\boldsymbol{V}_{bh12}, \boldsymbol{V}_{bh1} \cdot \boldsymbol{V}_{bh12}] \;\rightarrow\; [V_h, \theta_3, \theta_4] \tag{5.16}$$

これらのマップはどちらも (5.13) および (5.14) を構築したものと同じ取得データを用いて構築される．

目標の $[dt_{hr}, d\boldsymbol{p}_{bhr}]$ が与えられると，それに必要な打撃直後のボール速度 $\boldsymbol{V}_{bh2}$ が式 (5.15) の逆マップによって決まり，この $\boldsymbol{V}_{bh2}$ とマップ 1 によって予測された打撃直前のボール速度 ${}^*\boldsymbol{V}_{bh1}$ から打撃によるボール速度の変化 V_{bh12} が計算される．そして，このボール速度の変化 V_{bh12} を実現するような制御変数 V_h および (θ_3, θ_4) が，逆マップ (5.16) によって決定される．

ラケットの姿勢について

卓球台横方向へのボール変化に対応するためには，卓球ロボットの 4 自由度全てを用いて，打撃の瞬間のラケットの位置，姿勢，速度を適切に調整する必要がある．

ラケット表面の法線ベクトルは，モータ 3, 4 の回転角度 θ_3,θ_4 を用いて，$[cos\theta_4, sin\theta_3 sin\theta_4, cos\theta_3 sin\theta_4]^T$ と表される．つまり，ラケット姿勢とモータ回転方向の関係は，**表 5.1** のように表されるため，その方向だけとってみても，複雑な関係となる．

表 5.1　モータ 3，4 回転方向とラケット姿勢の関係

	$\theta_3 > 0$	$\theta_3 < 0$
$\theta_4 > 0$	Upper Right	Upper Left
$\theta_4 < 0$	Lower Left	Lower Right

一方，ラケット速度は，ラケット姿勢とは独立に制御することが可能であるが，ラケット表面法線ベクトルとラケット速度ベクトルとの成す角度が大きいと，打撃時のラケットとボールの摩擦から，打撃後のボールに複雑なスピンがかかり，ボール操作が困難になることが予想されるため，(5.17)) のような拘束条件を与える．

$${}^o\boldsymbol{V_r} = \begin{bmatrix} {}^oV_{rx} \\ {}^oV_{ry} \\ 0 \end{bmatrix} = \begin{bmatrix} \frac{C\theta_4}{\sqrt{C^2\theta_4 + S^2\theta_3 S^2\theta_4}} ||\boldsymbol{V_r}|| \\ \frac{S\theta_3 S\theta_4}{\sqrt{C^2\theta_4 + S^2\theta_3 S^2\theta_4}} ||\boldsymbol{V_r}|| \\ 0 \end{bmatrix} \tag{5.17}$$

マップ 2，マップ 3 の可視化と統合

図 5.12∼**図 5.15** は，入力座標格子点上の出力を LWR により推定し，直線補間することでマップを可視化したものである．**図 5.12** は，マップ 2 を可視化したものであり，$[V_{rx}, \theta_4] \rightarrow [v_{bh12x}, v_{bh12z}]$ の関係を示す．これは，計測したそのままの関係である．**図 5.13** は，そのマップ 2 の逆マップを示し，$[v_{bh12x}, v_{bh12z}] \rightarrow [V_{rx}, \theta_4]$ の関係を可視化したものである．こ

ちらが，実際の予測に用いる関係を示している．**図 5.14** は，マップ 3 の逆マップの可視化したものであり，$[dt_{hr}, dp_{bhrx}] \rightarrow [v_{bh2x}, v_{bh2z}]$ の関係を示している．

また，**図 5.15** は，打撃時のラケット状態決定に用いるマップであり，$[dt_{hr}, dp_{bhrx}] \rightarrow [v_{bh2x}, v_{bh2z}] \rightarrow [v_{bh12x}, v_{bh12z}] \rightarrow [V_{rx}, \theta_4]$ の関係を表す．ここで打撃直前ボール速度は，$v_{bh1x} = -4000[\mathrm{mm/s}], \mathrm{v_{bh1z}} = 2000[\mathrm{mm/s}])$ に固定している．

このように，飛行現象，反発現象をその状態量を用いた LWR により，滑らかな局面での近似的な表現が可能であるとわかった．ここで，LWR の距離関数の重み行列 $\boldsymbol{M}$ は I，重み関数の bandwidth は h=0.8 を用いた．また，打撃時刻，打撃時ボール状態予測と同様に各入力次元方向独立にデータベース内の統計値を用いて出力の分散値が 1 になるように標準化した上で LWR を行っている．この関係を用いて，望みの飛距離と飛行時間で打撃を行うためのパドル速度と姿勢を決定し，ボールコントロールタスクを実現する．

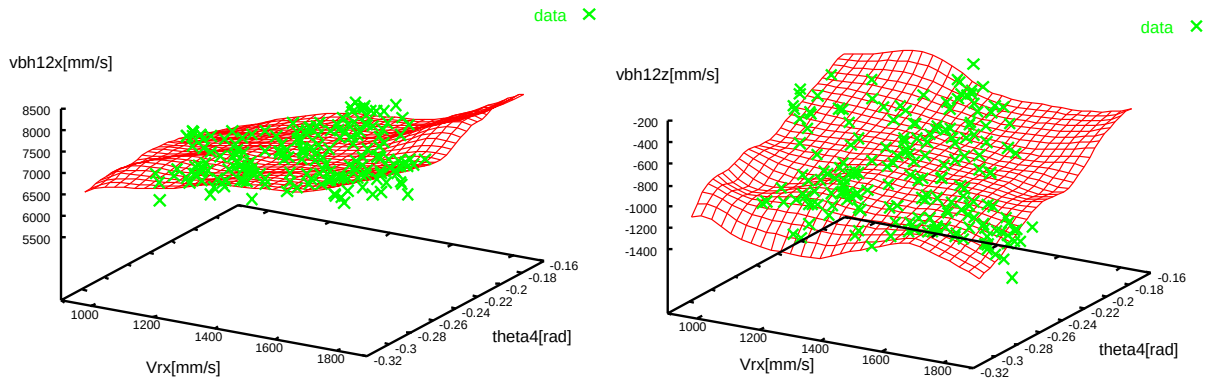


図 5.12 マップ $2(V_{rx}, \theta_4 \rightarrow v_{bh12x}, v_{bh12z})$

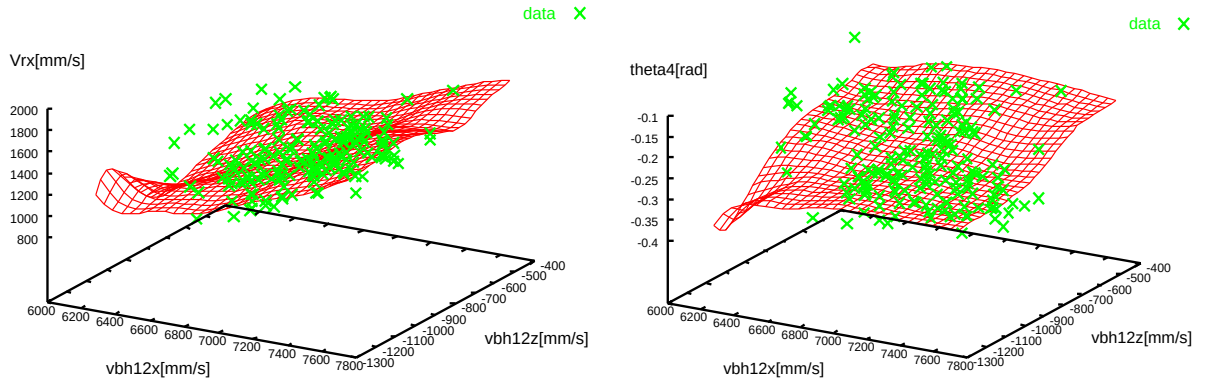


図 5.13 マップ 2 の逆マップ $(v_{bh12x}, v_{bh12z} \rightarrow V_{rx}, \theta_4)$

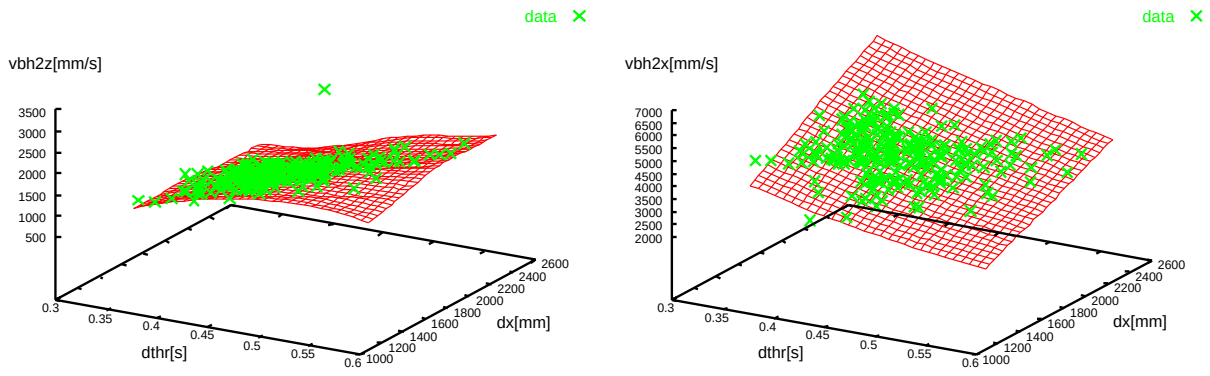


図 5.14　マップ 3 の逆マップ $(dt_{hr}, dp_{hrx} \rightarrow v_{bh2x}, v_{bh2z})$

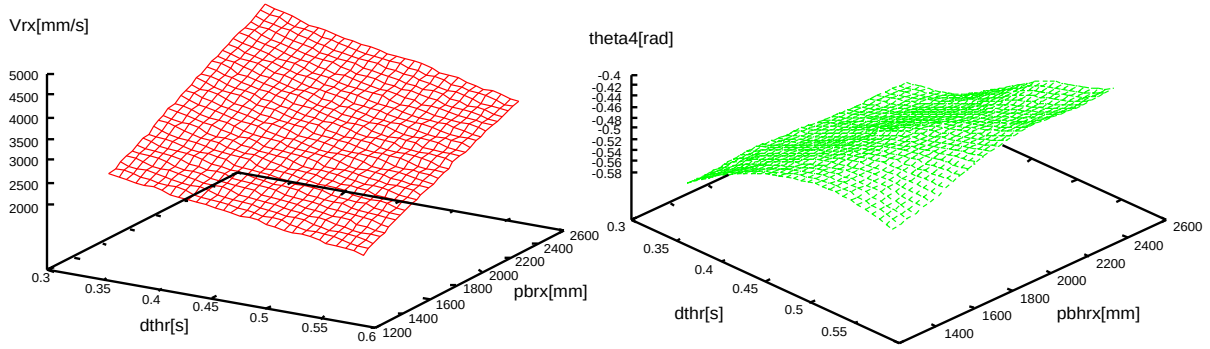


図 5.15　統合マップ $(dt_{hr}, dp_{hrx} \rightarrow V_{rx}, \theta_4$, $(v_{bh1x}$=-4000,v_{bh1z}=2000))

5.3.4　トレーニングフェーズ

まず，ボールの飛行現象，ラケットとの反発現象に関して何の知識もないロボットに，これらの現象を理解させるためのトレーニングフェーズとして，データベース構築を行う．このトレーニングフェーズでは，打撃時ラケット状態 (V_{rx}, θ_4) の予測は行わず，打撃時ラケット状態は $600 \leqq V_{rx} \leqq 1800[\mathrm{mm/s}]$，$-15 \leqq \theta_4 \leqq 15[\mathrm{deg}]$ で変化させて打撃を行い，その結果を計測する．トレーニングを行わなくても，予測，打撃を繰り返すことで，データベース構築が可能であるが，まず，データを幅広く取得し，あらかじめ全体の傾向を学習することにより，可動範囲外の大きく誤ったラケット状態を予測することを避け，結果として学習に要する時間を短縮することがトレーニングフェーズの狙いである．

図 5.16 はトレーニングフェーズでの各試行毎の状態量を表す．ここで，打撃直前ボール速度 $\boldsymbol{v}_{bh1}$ は予測値である．この図からラケット姿勢 θ_4 を大きくすればラケットが上に向き，打撃後ボールの z 方向速度 v_{bh2z} も上昇し，結果として飛行時間 dt_{hr} が長くなるといった当り前の傾向が見られる．ロボットはこのトレーニングフェーズで，人間にとっては当

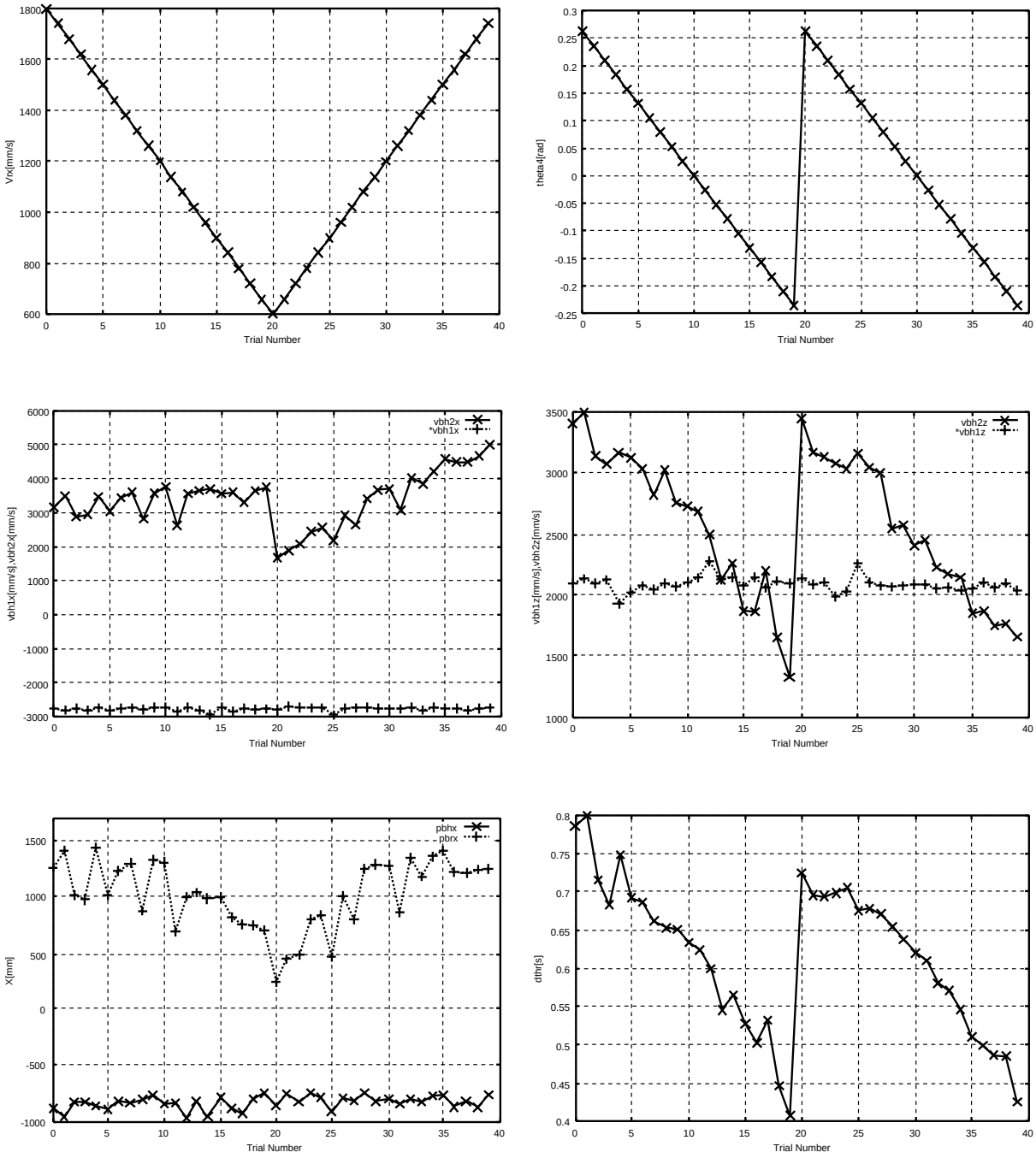


図 5.16 トレーニングの様子

然と思われる物理現象を学習することにより，おおまかな予測が行えるようになる．こういった粗い物理モデルで表現できる傾向以外の現象も学習することを期待し，LWR 学習を用いる．

5.3.5　LWR のチューニング

飛行モデル，打撃モデル共に精度よく予測を行うためには，LWR のパラメータチューニングを行うことが望ましいと思われる．しかし，連続した打撃中でのオンラインチューニングは，処理コストが大きくなるため，ここでは，トレーニングデータを用いたオフラインでのチューニングを行うこととした (付録 A.2.2 参照)．

表 5.2 に，飛行モデルをチューニングした結果を示す．z 方向速度と飛行時間の関係が特に大きな重みとなっていること，x, y 方向速度に関しては，それぞれの方向の飛行距離と飛行時間とに距離重みをおいた予測を行うことが有効であることを表している．このことから，重みの小さくなったパラメータそのものをモデルより除外することも有効であるように思える．しかしながら，一般的に飛行時のボールには，重力，速度方向と逆向きの摩擦力，および飛行速度とスピン双方に応じたマグヌス効果による力が働くとされていることから，パラメータ数を減らすことが有効とは一概には言えない．そこで，飛行マップの入力数を減らさずに，**表 5.2** の距離関数の重み行列 M，重み関数の bandwidth h を用いることとする．

表 5.2　m_{jj} of Distance Function &Bandwidth h

	dt_{hr}	dp_{hrx}	dp_{hry}	h
v_{bh2x}	2.405	0.575	0.183	0.403
v_{bh2y}	1.238	0.004	2.901	0.317
v_{bh2z}	1.498	0.000	0.000	0.668

飛行マップの様子

図 5.17～**図 5.19** は，トレーニングフェーズで取得されたデータを用いて，格子点上の出力をチューニング済の LWR により予測し，直線補間したものである．

チューニング済マップの例として，**図 5.17** に，X 方向のマップ 3 の逆マップ $((dt_{hr},dp_{hrx})\rightarrow v_{bh2x})$ を示す．ここでは，dp_{hry}=0[mm] のみ表示したが，dp_{hry}=500[mm],-500[mm] においても，このマップにほとんど一致する．**図 5.18** は，Y 方向のマップ 3 の逆マップ $((dt_{hr},dp_{hrx})\rightarrow v_{bh2y})$ を表し，**図 5.19** は，Z 方向のマップ 3 の逆マップ $((dt_{hr},dp_{hrx})\rightarrow v_{bh2z})$ である．ここでも，dp_{hr}=0[mm] 以外の場合は，このマップにほとんど一致するため省略した．

図 5.20 では，Y 方向のマップ 3 の逆マップを，グローバルにチューニングしたものとしなかったものとを比較した．チューニングしたものは，飛行時間 dt_{hr} の変化に敏感に対応している一方で，チューニングしなかったものは，dt_{hr},dp_{hry} のみならず，dp_{hrx} の距離も参照し重み付けをするため，dt_{hr} の変化に対応できないことがわかる．

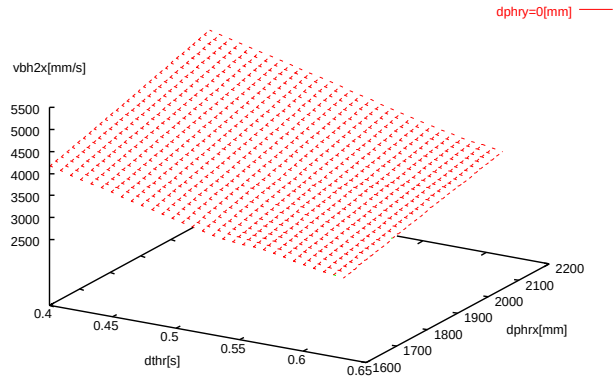


図 5.17 $dt_{hr} - dp_{hrx} - v_{bh2x}$

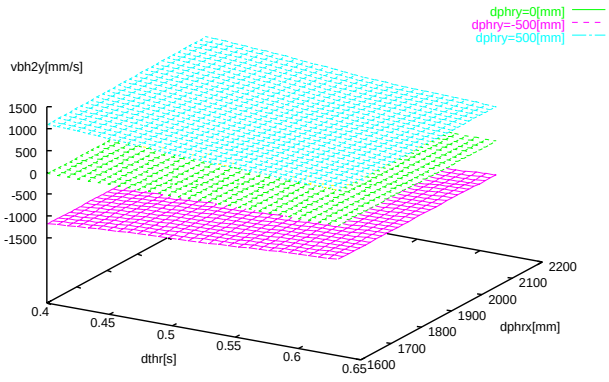


図 5.18 $dt_{hr} - dp_{hrx} - v_{bh2y}$

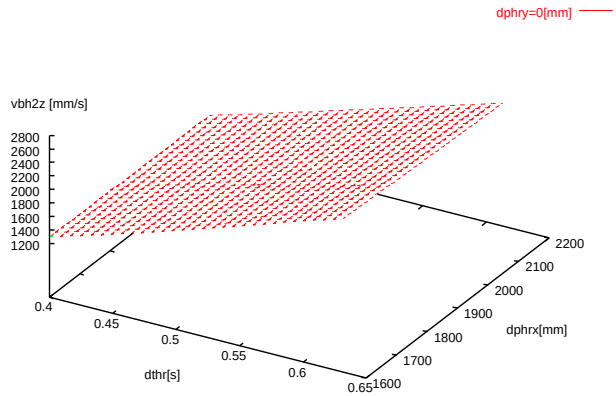


図 5.19 $dt_{hr} - dp_{hrx} - v_{bh2z}$

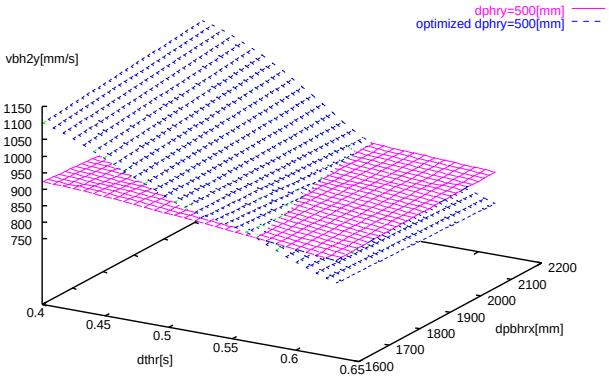


図 5.20 $dt_{hr} - dp_{hrx} - v_{bh2y}$

打撃マップの様子

図 5.21 から**図 5.23** は，トレーニングフェーズで取得されたデータを用いて，格子点上の出力を LWR により予測し直線補間したものである．ここでは LWR のチューニングは行わず，距離関数の重み行列 M=I，重み関数の band width h は 0.5 とした．

ここでも，飛行マップ同様にマップの様子を可視化するために，v_{bh12z} に関しては，0,500,-500[mm/s] の 3 通りの離散値とした．まず，**図 5.21** はラケットスピードのマップ 2 の逆マップ $(v_{bh12} - ||V_r||)$ を 2 方向から表したものである．また，**図 5.22** はラケットの 4 軸角度のマップ 2 の逆マップ $(v_{bh12} - \theta_4)$ を 2 方向から表したものである．v_{bh12z} が大きい程ラケットを上に向け，逆に v_{bh12x} が大きいほど，ラケットを下に向けるという傾向が表れている．**図 5.23** は，ラケットの 3 軸角度のマップ 2 の逆マップ $(v_{bh12} - \theta_3)$ である．複雑な形となっているが，θ_4 のマップと合わせて見ると，**表 5.1** に表したラケット方向とモータ回転角度の関係を示している．しかし，v_{bh12y} が 0 付近では，ラケット姿勢を正反対に予測してしまう可能性があることがわかる．これは卓球ロボットの姿勢制御機構の問題だと言えるが，近傍点を参照して予測を行う LWR の欠点とも言える．

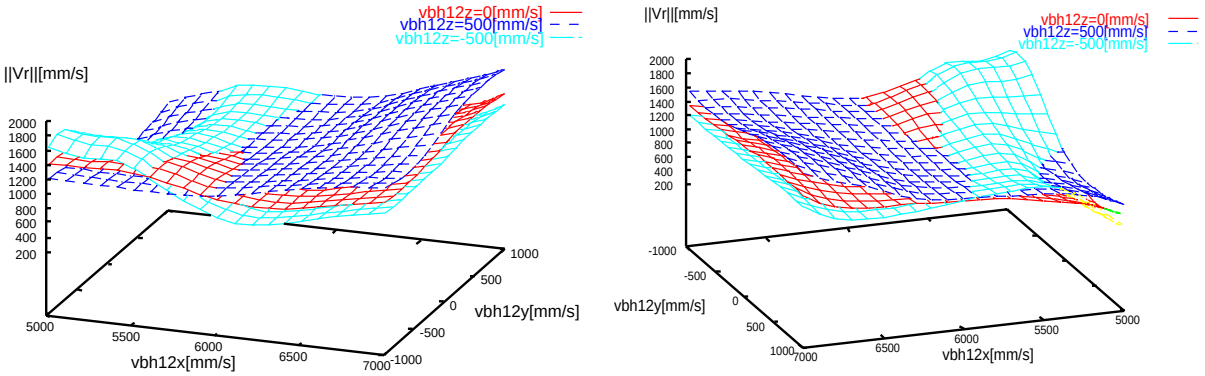


図 5.21　$\boldsymbol{v}_{bh12} - ||V_r||$

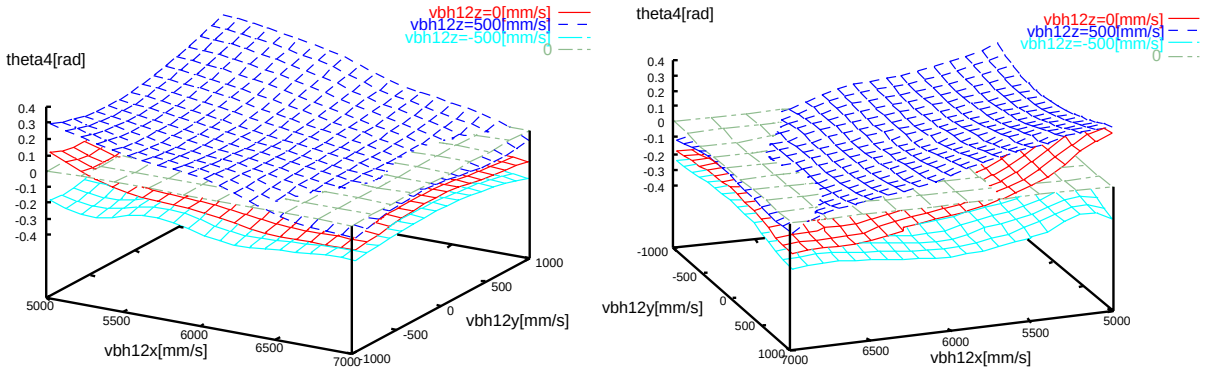


図 5.22　$\boldsymbol{v}_{bh12} - \theta_4$

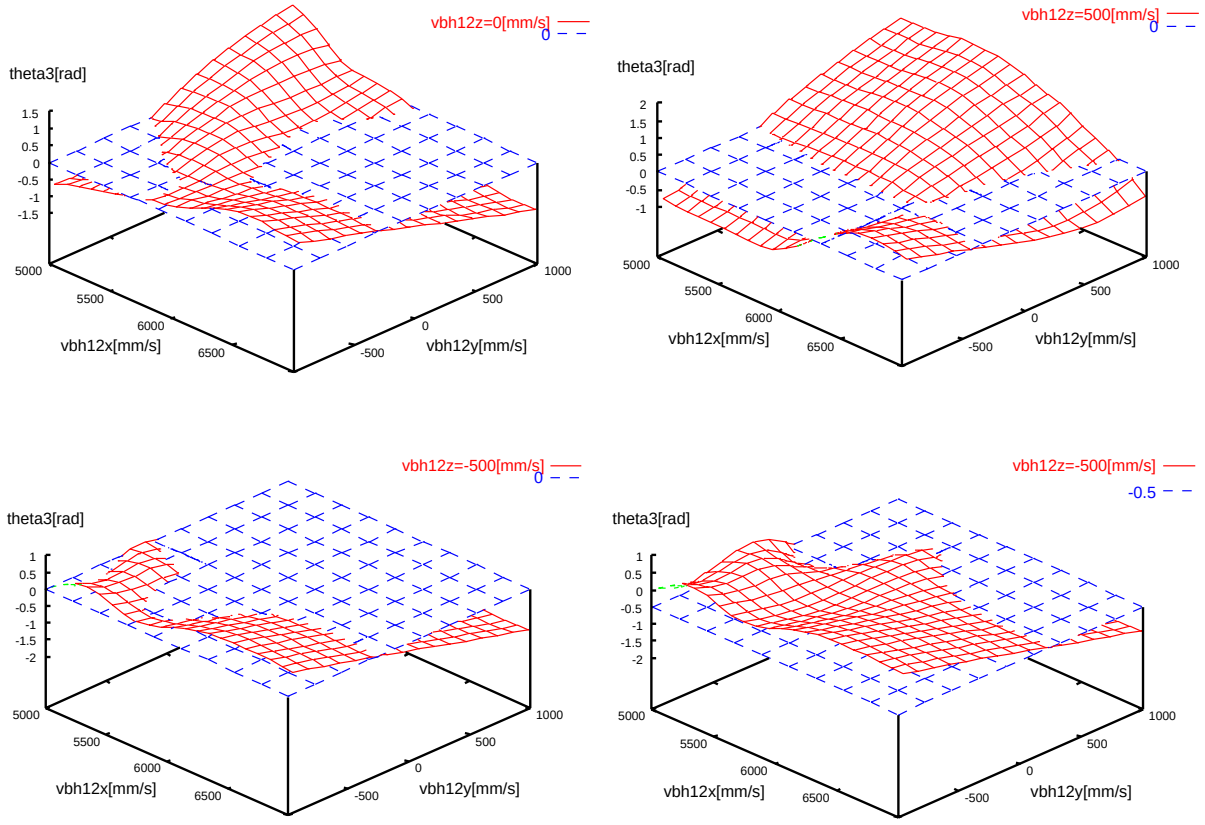


図 5.23 $\boldsymbol{v}_{bh12} - \theta_3$

5.4　ボールコントロール検証実験

前節で提案した「マップを用いた打撃時ラケット状態予測」が実現可能か確認するための実験を行った．本実験では，打球機から供給されるボールを，目標 (飛行時間 dt_{hr}，落下位置 p_{brx}) 通りに打ち返すことができるラケット状態を LWR 学習により獲得することを目標とする．ボールを供給する打球機の設定は，同一スピン，同一打ち出し角度とし，速度は強弱 2 パターン設定している．また，打撃時刻，打撃時ボール状態予測に用いるデータベースには，実験と同一設定の打球機から供給されるボールデータから抽出したデータ 200 点を用いている．

実験は，5.3.4 節のトレーニングフェーズ終了時から，目標飛行時間および目標落下位置を設定した打撃を開始する．ただし打球機の設定は，前節同様に打ち出し角度，速度，スピンを固定した．

5.4.1　飛行時間の操作

目標返球位置：p_{brx}=1100[mm],p_{bry}=300[mm]
目標飛行時間：dt_{hr}=0.5[s]，dt_{hr}0.7[s] を交互 に指定

図 5.24 は，飛行時間 dt_{hr}，落下位置 p_{brx},p_{bry}，打撃時ラケット速度 V_{rx},V_{ry}，及び打撃時ラケット姿勢 θ_3,θ_4 の試行毎の推移を表し，横軸は試行回数である．ここで，試行番号 0 から 119 までは前節で示したトレーニングフェーズであり，図からは省略した．**図 5.24** で，試行回数 150，つまり目標を設定した打撃開始後 30 回付近で，急にラケット速度 V_{rx}, V_{ry}，モータ角度 θ_3, θ_4 の予測値が変化しており，試行開始時よりも飛行時間 dt_{hr} が両目標共に上昇し，ほぼ目標値を実現していることがわかる．これは，学習により目標を実現する状態近辺のデータ密度が上昇し，マップが変化した影響であると思われる．**図 5.25** は，トレーニング終了時の反発マップと試行回数 147 回時の反発マップを部分的に可視化したものである．この図のように，データ密度が上昇した領域のマップが更新されることで類似入力に対する予測値も大きく変化する．

図 5.26 は，トレーニング終了時の反発マップと，トレーニング終了からさらに 46 回学習後の反発マップの全体図を示す．前述のとおり，データ密度の上昇した付近のマップ形状が大きく変化している．マップを更新することによる学習の効果が現れていると言える．

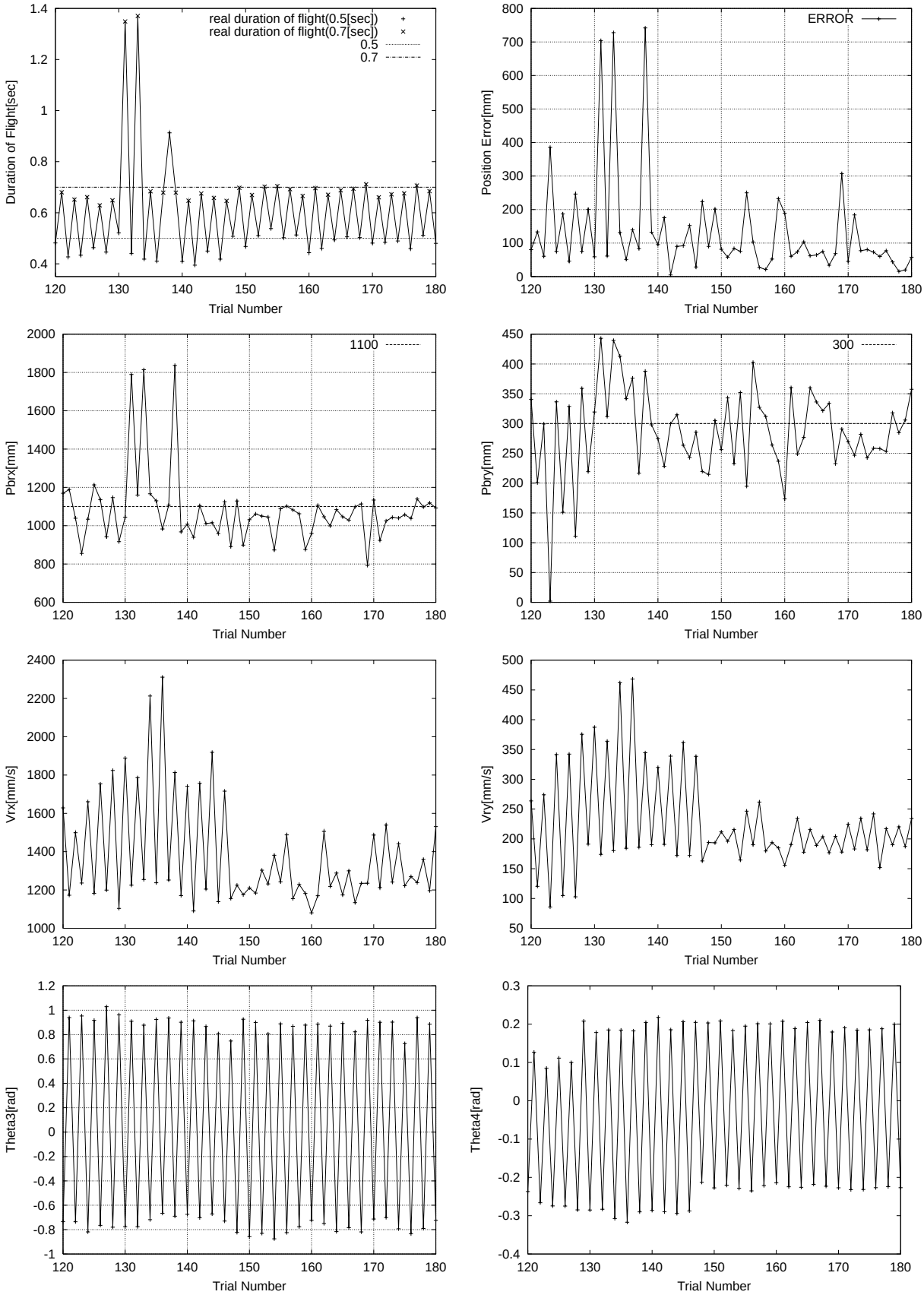


図 5.24 飛行時間操作の学習過程

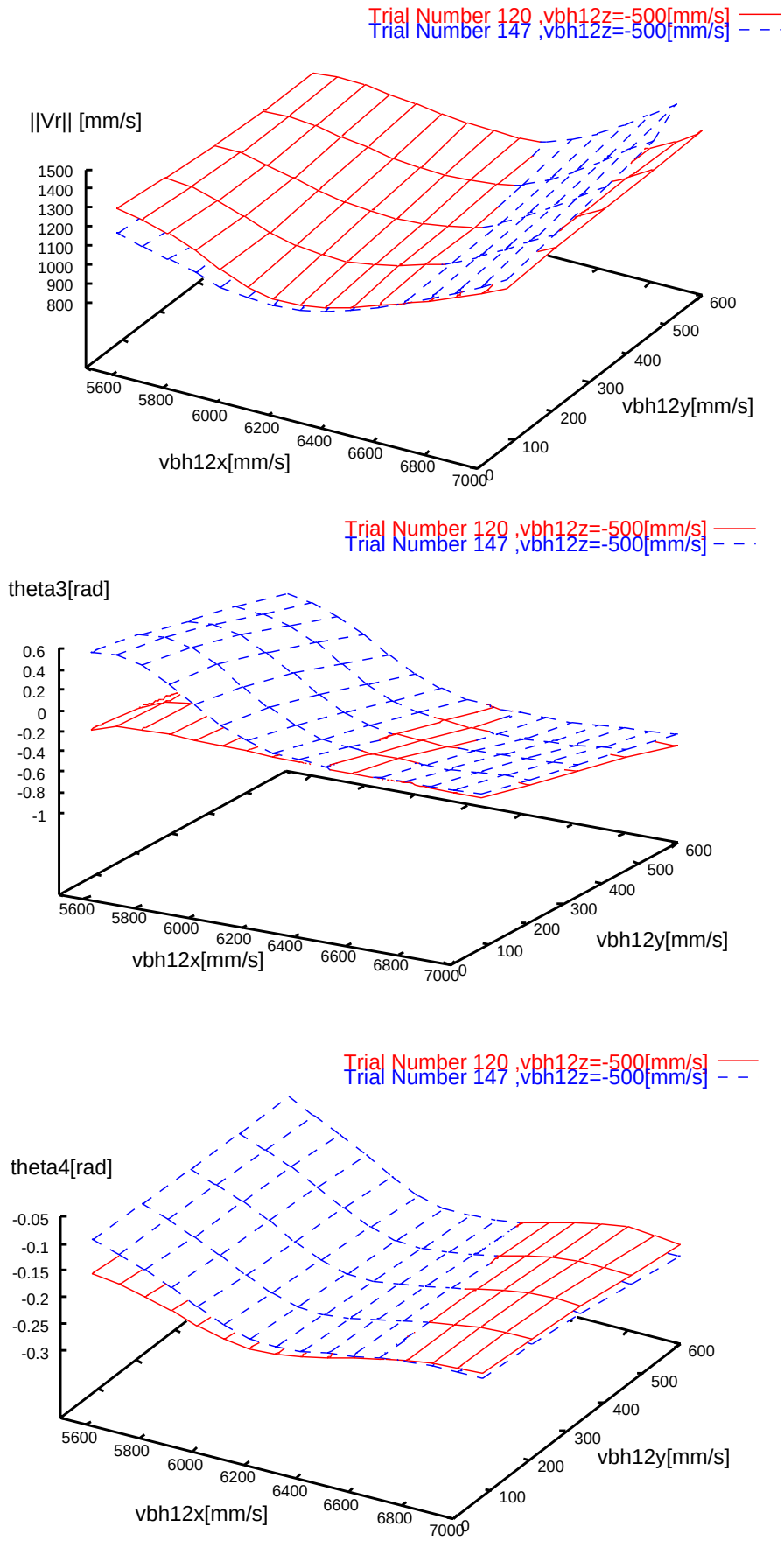


図 5.25　Transformation of Hitting Map(Local View)

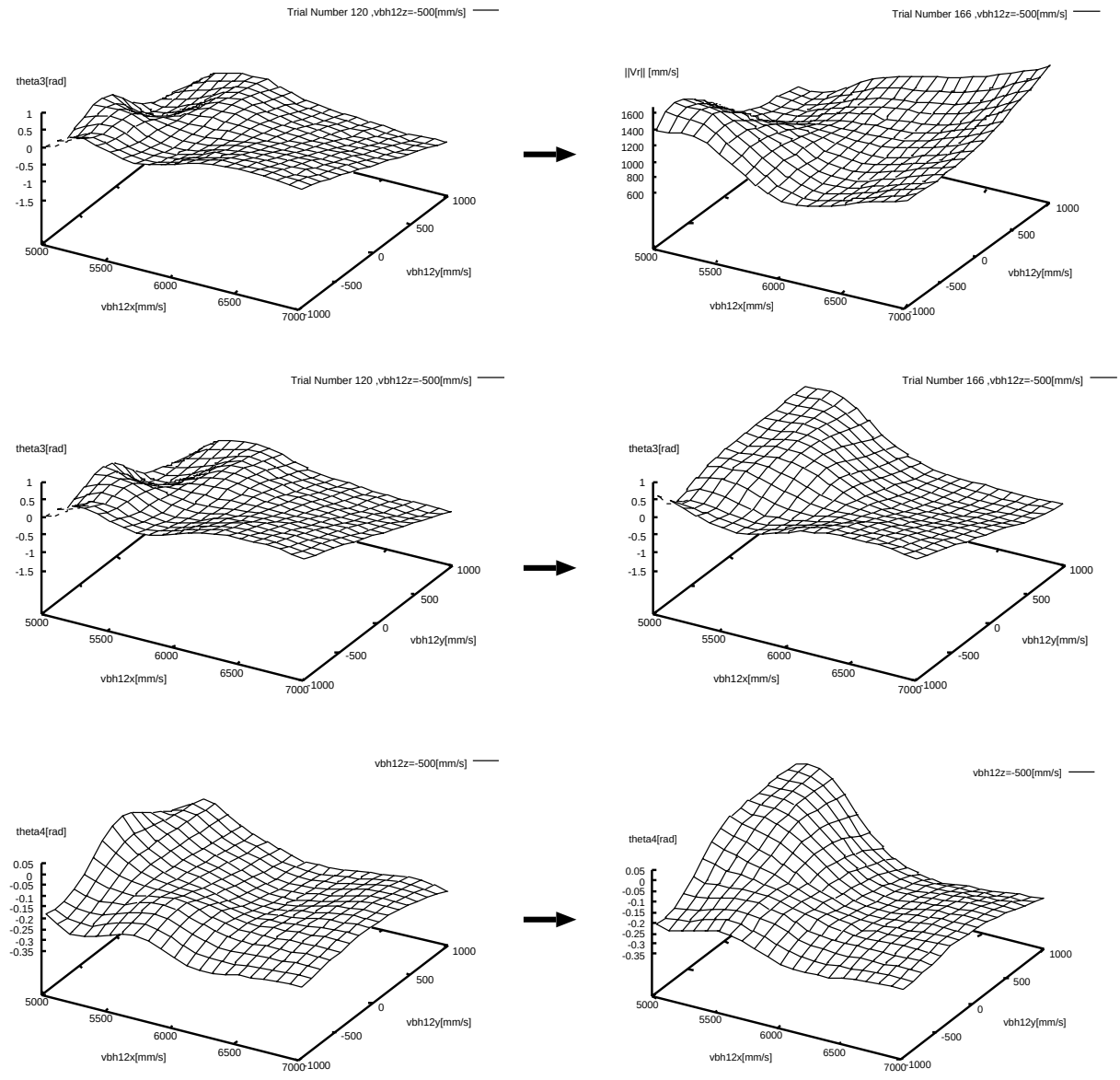


図 5.26 Transformation of Hitting Map(Global View)

図 5.27 から**図 5.30** は，十分学習後 (179 回目と 180 回目) の目標飛行時間 dt_{hr} の異なる打撃の様子を卓球台座標系で表したものである．**図 5.27**，**図 5.28** は，そのボール軌道を比較したものであり，**図 5.29** は $dt_{hr} = 0.7$[s] でのボールとラケット位置変化及び，ラケット速度，姿勢の時間変化を表す．**図 5.30** は同様に $dt_{hr} = 0.5$[s] での変化を表す．

双方ともに，x 座標-1800[mm]，y 座標 0[mm]，z 座標 300[mm] 付近から打球機により打ち出されたボールが，x 座標 600[mm] 付近で卓球台で跳ね返り，x 座標 800[mm] 付近でラケットにより打撃され，目標打ち返し位置である x 座標-1100[mm]，y 座標 300[mm] に向けて飛行を開始している．打球機から打ち出されたボールに若干の違いがあるが，打撃後のボール軌道を比較すると，飛行時間を目標通りうまく操作し，目標位置付近にボールを落下させることに成功している．

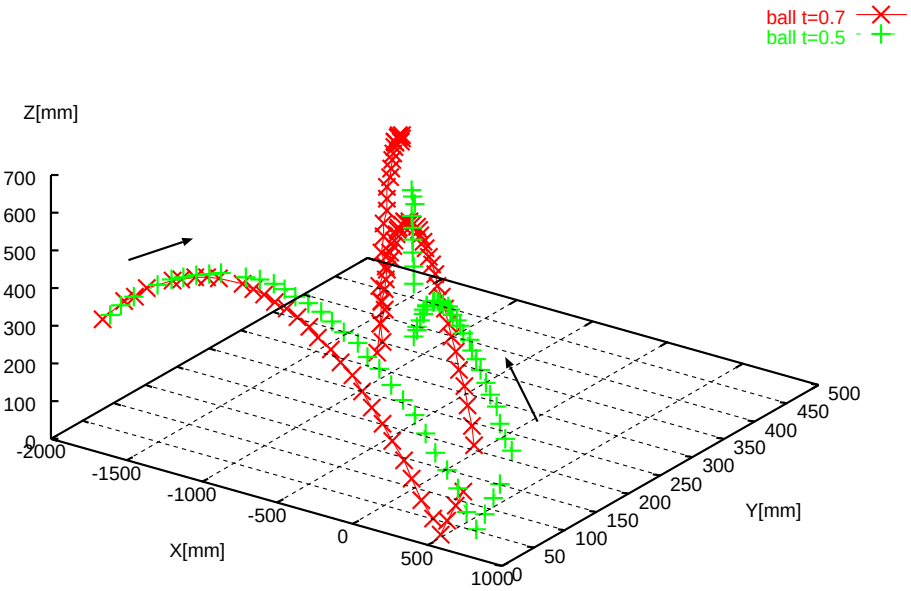


図 5.27　Ball Trajectory ,x-y-z

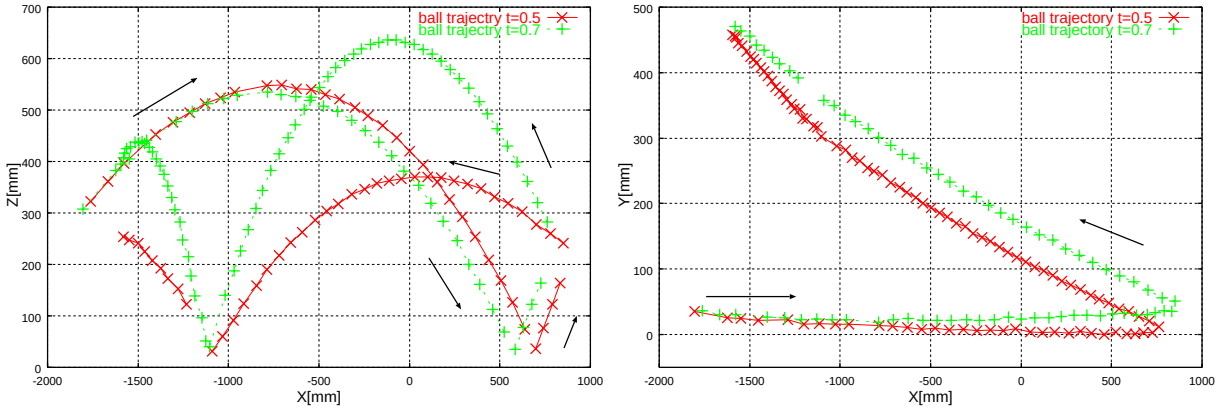


図 5.28　Ball Trajectories ,$x-z$,$x-y$

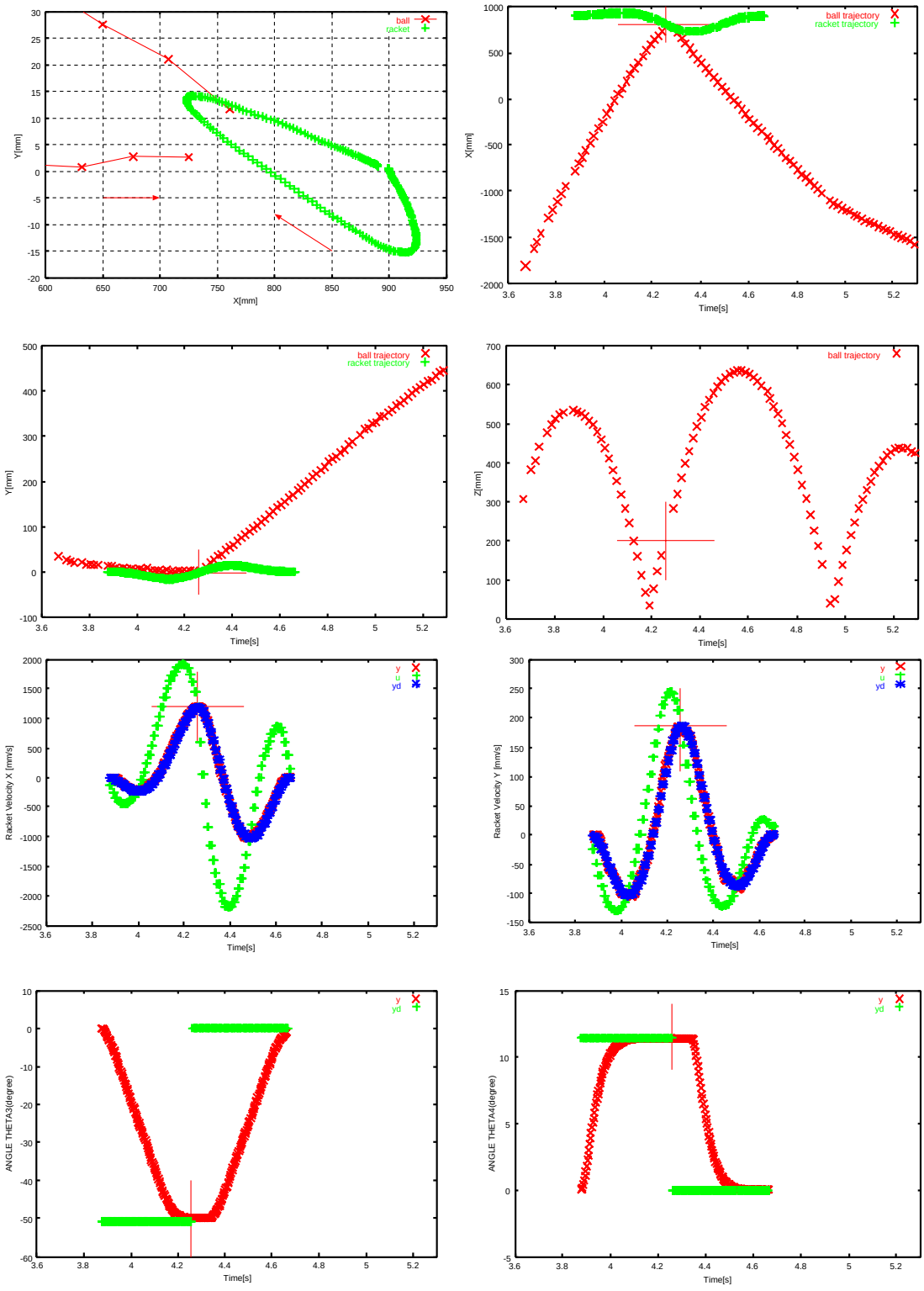


図 5.29 Ball & Racket Statuses (Trial No.179)

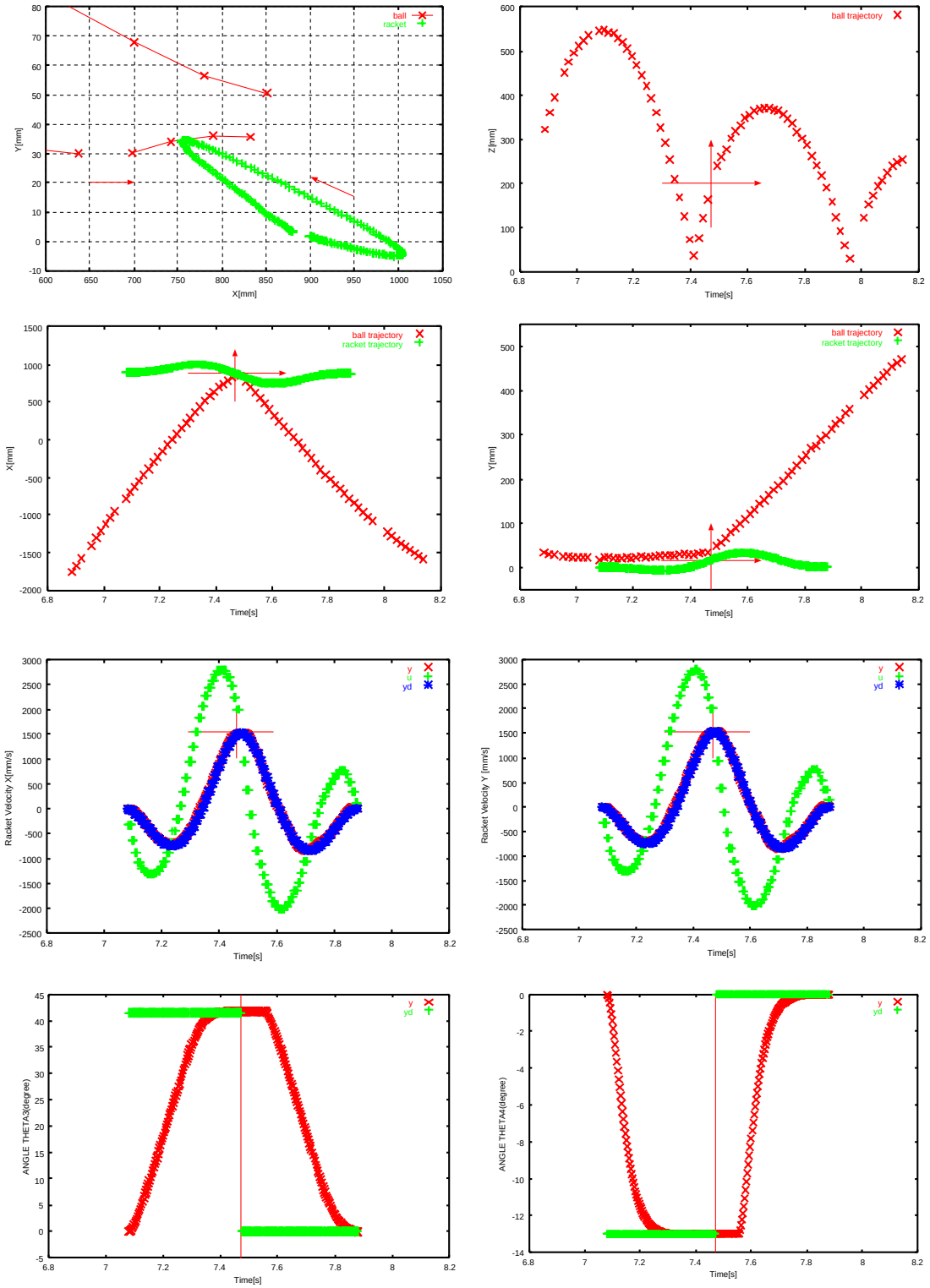


図 5.30　Ball & Racket Statuses (Trial No.180)

5.4.2 落下位置の操作

飛行時間の操作と同様に目標落下位置を変化させて実験を行った．

目標返球位置：p_{brx}=900[mm], p_{bry}=300[mm]，-300[mm] を交互 に指定
目標飛行時間：dt_{hr}=0.4[s]

図 5.31 は，このタスクでの各試行毎の状態遷移を表す．このタスクでも，目標を設定した打撃開始後 10 回程度学習した後に急に予測値が変化し，目標飛行時間 dt_{hr}，目標落下位置 $\boldsymbol{p}_{br}$ をほぼ実現していることがわかる．

計測誤差が打ち返しボール落下位置に与える影響を評価した．QuickMAG の計測誤差が位置にしておよそ 5mm 程度であり，最小二乗法を用いる本手法では位置誤差は 2.5[mm] 程度，速度は 95[mm/s] 程度に抑えられる．ボールが計測平面を通過してから，ロボットに打ち返されて 0.8[s] で再び相手コートで跳ねると仮定すると，計測誤差に起因する返球位置の誤差は標準偏差でおよそ 80[mm] と推定される．**図 5.32** に示したロボットによる返球位置誤差はこの値に近く，仮想ターゲットを決定したマップの有効性を示すものである．

10 年程度の卓球経験の有る人間のプレーヤに同様のタスクを行ってもらい，その精度を計測した．**図 5.33** が 40 回の試行を行った際の落下位置の誤差である．標準偏差にして x,y 方向それぞれ，111[mm]，67.1[mm] の誤差であった．これより，ロボットがほぼ人と同等の精度で打ち返しを実現したことがわかる．また，人もロボットも目標位置を越えて打撃を行う傾向が見られた．

5.5 対人ラリータスク

5.5.1 はじめに

前節までに，マップによる打撃時ボール状態予測 (5.3.1 節)，LWR 学習による打撃時ラケット状態決定 (5.3.3 節) と DirectILC によるラケット動作 (4.2 節) を実現し，これを組み合わせて飛来するボールを飛行時間と落下位置を指定して打ち返すボール操作タスクを実現したので，その応用として対人ラリータスクを行った．

ここで，対人ラリータスクとは一般的に人間同士で行われている卓球のラリーを意味し，人間が打撃するボールをロボットが打ち返し，そのボールを人間が再び打ち返すという作業を繰り返す．これは，前節のボールコントロールタスクの繰り返しと考えられる．

ロボットのラケットは卓球台上を移動するため，対戦相手との距離が実際の卓球環境より短くなる．そこで実験環境は，**図 5.34** のようにロボット側卓球台と人間側卓球台の間を 300[mm] 空け，人間側卓球台の端にネットを設置する事により実際の卓球環境に近づけた．また，ロボットの可動範囲，打撃可能範囲を考慮して，次のような条件を設定し，この条件を満たさない場合は打撃を行わない事とした．

- 打撃位置，打撃までの時間に関する条件
  - dt_{ih} ≧0.365[s], -1080.≦ p_{bhx} ≦-500.[mm] ,$|p_{bhy}|$ ≦400.[mm]
- 動作軌道の極値に関する条件
  - $|V_{rx_{max}}|$ ≦ 3000.[mm/s], -1120. ≦ $p_{rx_{max}}$ ≦ -350.[mm], $|V_{ry_{max}}|$ ≦ 1800.[mm], $|p_{rx_{max}}|$ ≦ 450.[mm/s] ,$|a_{r_{max}}|$ ≦ 25000. [mm/s^2]

このように，ロボットの打撃可能範囲が狭いため，卓球ロボットにとっても，対戦者にとってもラリーを継続させることは困難となる．ロボットにとっても，対戦者にとっても，勝つための卓球をすることは容易であるが，ラリーを継続させるためには，対戦者がロボットにとって打ち返しやすいボールを供給すること，ロボットが対戦者にとって打ち返しやすいボールを打ち返すことが求められるため難しくなる．ロボットは，打撃後のボールが落下するまでの時間 dt_{hr} と落下位置 $\boldsymbol{p}_{br}$ を操作し，対戦者にとって打ちやすいボールを返球する必要がある．

5.6　実験手順

前節のボール操作タスク同様に，以下の流れで学習を行う．

① 対戦者が一人で打つサービスボールの軌道を計測する．ただし，本実験でのサービスは卓球ルールのものとは異なり，自コートでバウンドさせずに直接相手コートに落下させる．また，ボールを対戦者に向かって投入し，対戦者によって打撃された軌道も計測する．

② ボール軌道データから打撃時刻，打撃時ボール状態予測のマップ (マップ 1) を作成する．

③ 対戦者がサービスしたボールをマップ 1 を用いて予測された打撃時刻，および位置で打撃する．ただしこの時のラケット状態 (速度，姿勢) は予測せずに指定値を用いる．

④ 上の打撃結果を計測し，打撃時ラケット状態予測マップ (マップ 2，マップ 3) を作成する．

⑤ 目標となる打ち返し位置 $\boldsymbol{p}_{br}$，飛行時間 dt_{hr} を設定し，対戦者が打ち出したボールから，マップ 1，2，3 を用いて，打撃時のラケット状態 $(||\boldsymbol{V}_r||, \theta_3, \theta_4)$ を決定して打撃動作を行う．そしてその結果を計測し学習する．

5.7　対人ラリータスク実験

目標飛行時間 dt_{hr}=0.55[s]，目標打ち返し位置を x 方向は p_{rx}=-1550[mm],y 方向は落下後上昇時に卓球台中心 y=0[mm] を通過するように (対戦者が打ちやすいように)，p_{ry}=0.3×$^*p_{bhy}$

に設定した．ここで，${}^{*}p_{bhy}$ は y 方向の予測打撃位置である．マップ 1 は，ロボット側から人間が投げたボールを対戦者が打ち，そのボールを計測することで，ラリー時のボール軌道を模したマップデータを取得しておいた．

このような条件でラリータスクを行った結果，平均で 5 回程度，最大 14 回のラリータスクを行う事ができた．**図 5.35** に実験の様子を示す．

図 5.36〜**図 5.43** は，14 回継続時のラリーの様子を表す．**図 5.36** は，x,y 方向のボールの時間変化及び，ロボットによる打撃位置 (p_{bh}) とボール落下位置 (p_{br}) を示す．**図 5.38** は，x-y 平面でのラケット中心の軌跡を表す．ここで，ラケットの待機位置は x=900[mm],y=0[mm] である．**図 5.39** は，各打撃による打撃後ボールの飛行時間 (dt_{hr}) 及び，落下位置 (p_{br})，落下位置の誤差を表す．打ち返しの精度はよくないが，ラリー継続に問題がない (対戦者がカバーできる) 程度である．

図 5.40, **図 5.41** はラリー開始から 11[s] までのボールとラケットの位置変化及びラケット速度，姿勢変化を表す．ここで，“TASK A,B” は打撃タスク中，“TASK C” は待機タスク中であることを示す．また，$racket$ はラケット中心位置，p_{bhr} はロボットによる打撃位置，${}^{*}p_{bhr}$ は予測打撃位置，p_{br} は打撃後のボール落下位置，Vr はラケット速度，Vrh は打撃時ラケット速度の指令値，$theta3, 4$ はラケット姿勢を決定するモータ 3，4 の角度，$theta3\_h, theta4\_h$ は打撃時のモータ 3，4 の指令角度を表す．打撃位置の予測値 ${}^{*}p_{bh}$ と実際の打撃位置 p_{bh} が 50[mm] 程度ずれており，おそらく打撃時ボール速度の予測もこれに相応する程度ずれていると予想できる．打撃時刻に関してはあまりずれはないが，実際の打撃時刻におけるラケット状態は予測状態とは異なっていると予想される．こういった誤差全てが，打撃後のボール軌道に影響するため，ロボットによる打撃後の落下位置 p_{br} はばらついていることが $x-z$ 方向のボール軌跡からわかる．しかし，対戦者にとって打ちやすい位置へとボールを打ち返せていることが**図 5.40** の x-z 方向のボール軌跡からわかる．**図 5.40** の x-y 平面でのボール軌跡をみると対戦者が y 方向プラス側へと大きく打ち返したボールをロボットは卓球台中心付近に打ち返している．**図 5.42**, **図 5.43** は，11[s] 以降のラリー後半の様子であり，前半と大差がないため詳細の説明は省く．最後の 1 回は y 方向の打撃位置予測誤差が大きく，ラケットの端にボールが当たり，y=-400[mm] にボールが落下した．このため，対戦者が打ち返したボールが計測平面を通過せず，予測が行えなかったためにロボットが動作することなくラリーは終了した．

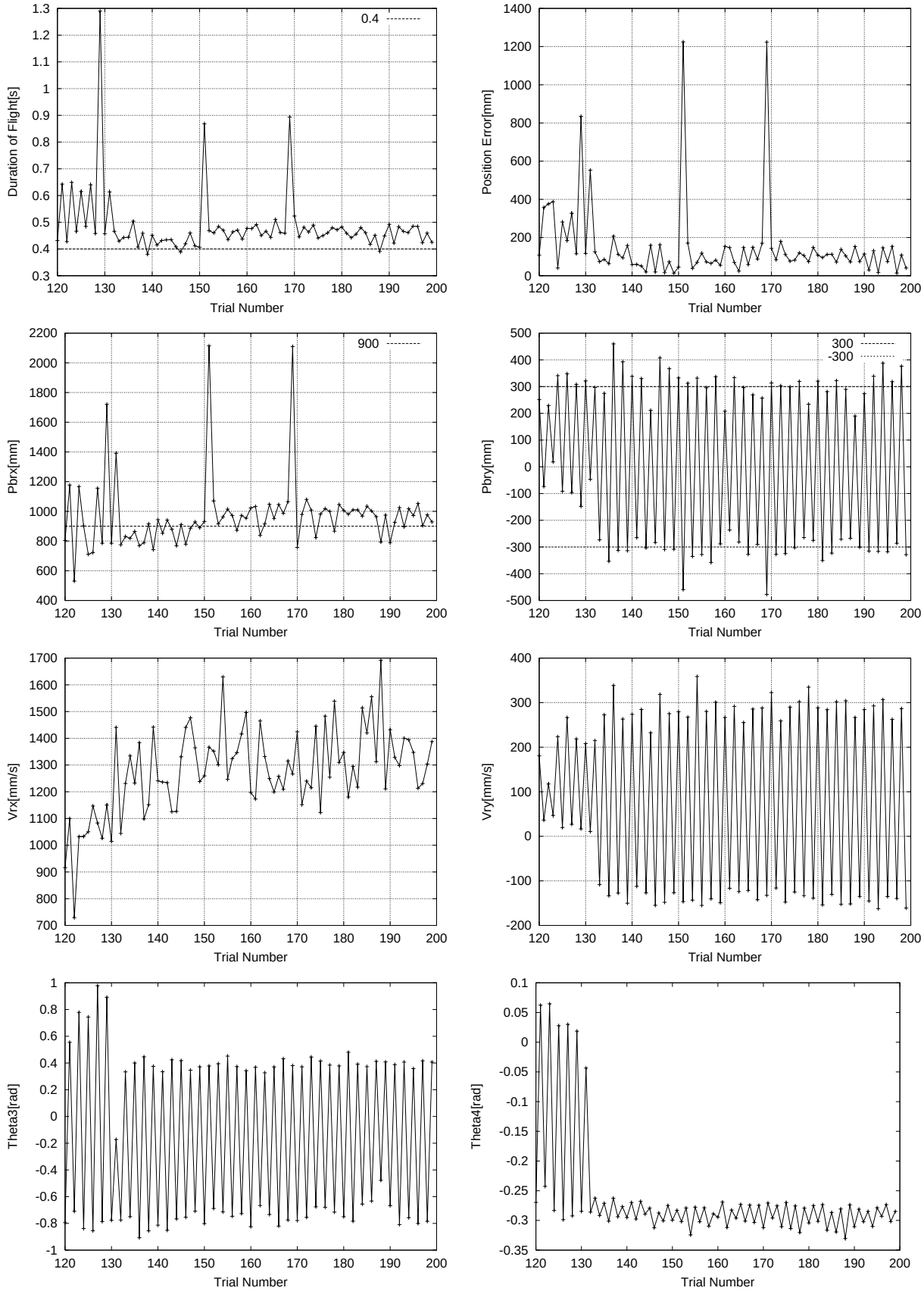


図 5.31 Landing Position Control

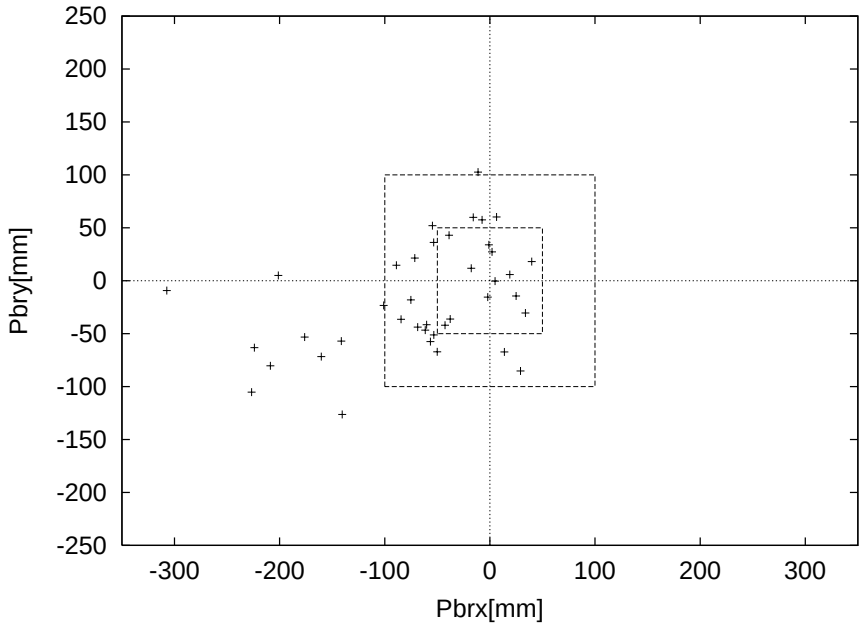


図 5.32 Errors in the landing point by a robot

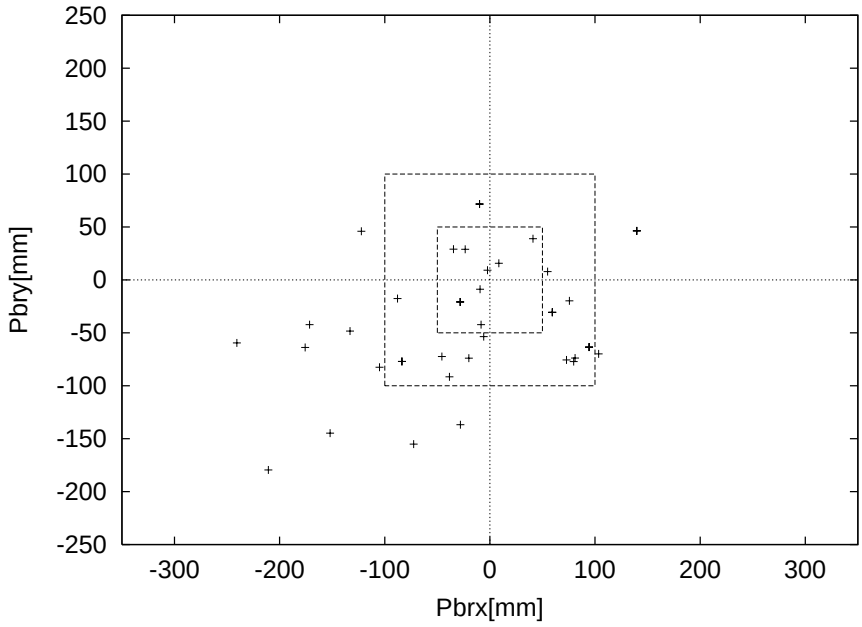


図 5.33 Errors in the landing point by a human

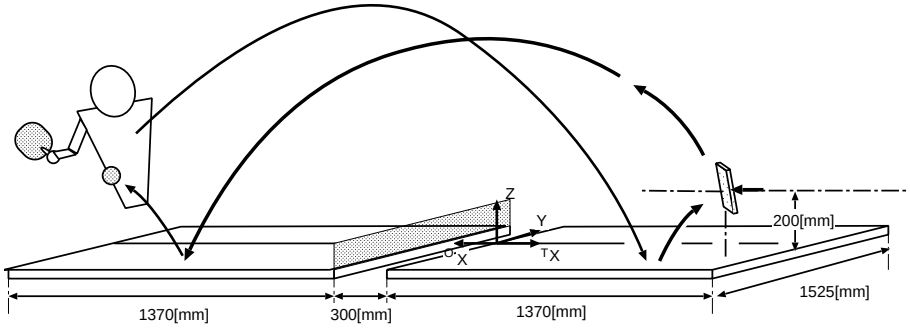


図 5.34　対人ラリータスクの実験環境

図 5.35　対人ラリータスク実験の様子

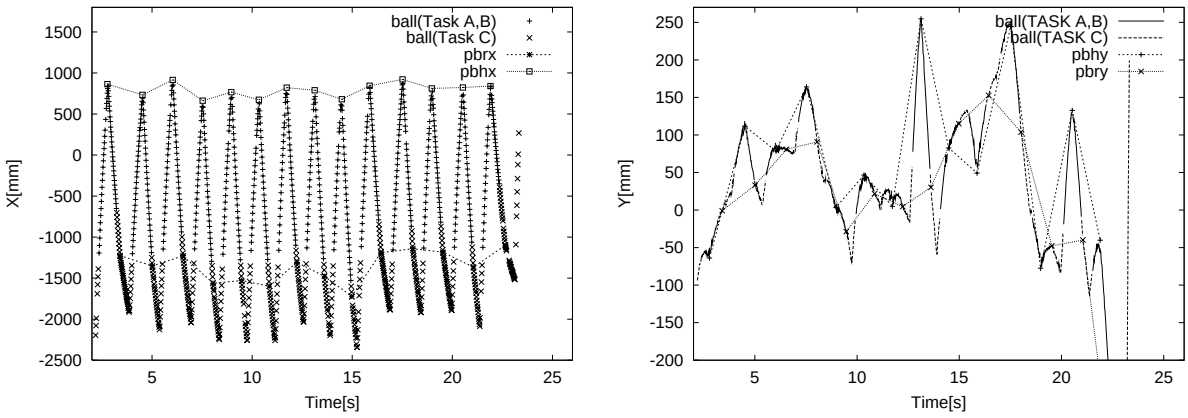


図 5.36　Ball Trajectories with Hit & Round Positions

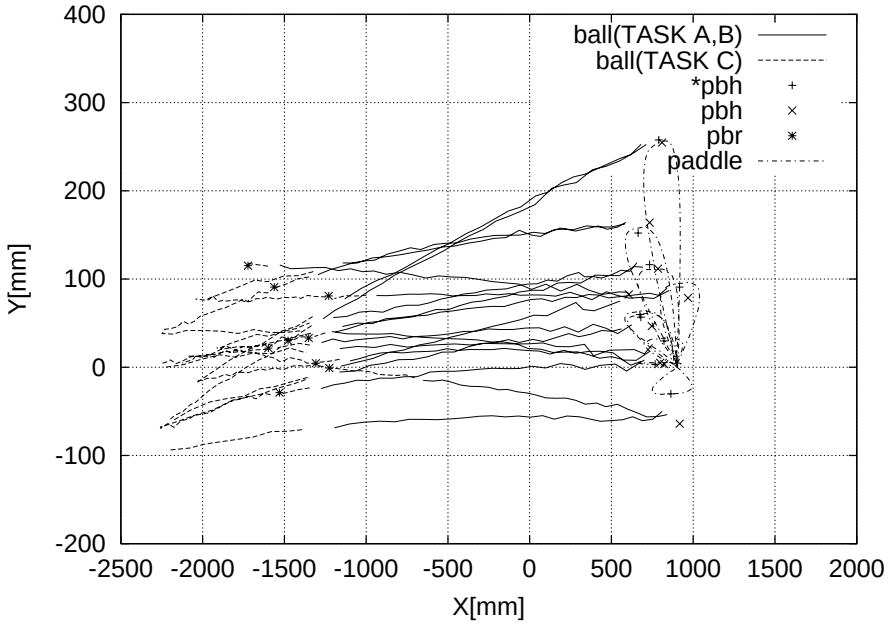


図 5.37 Ball trajectory in the "rally task" (in $X-Y$ plane)

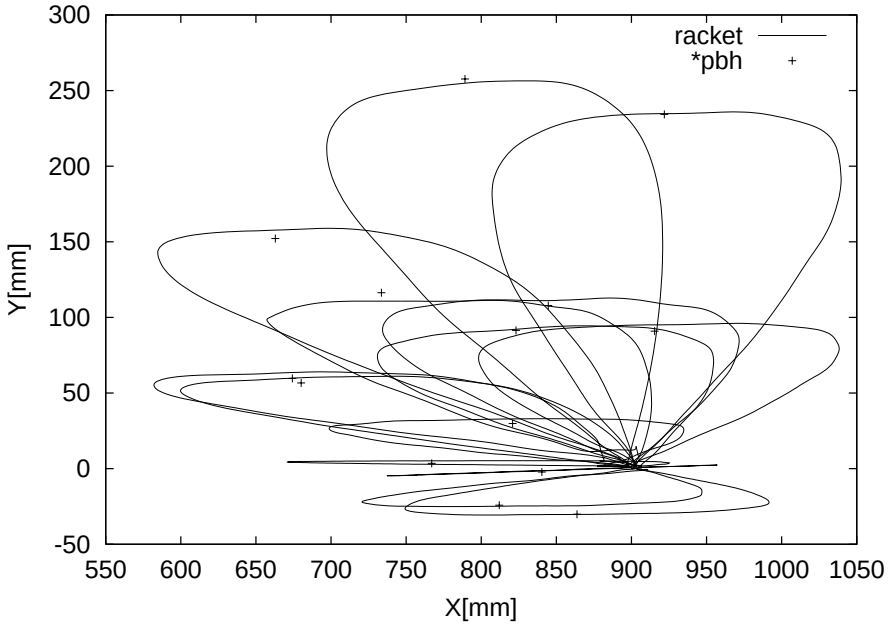


図 5.38 Paddle trajectory x in the "rally task"

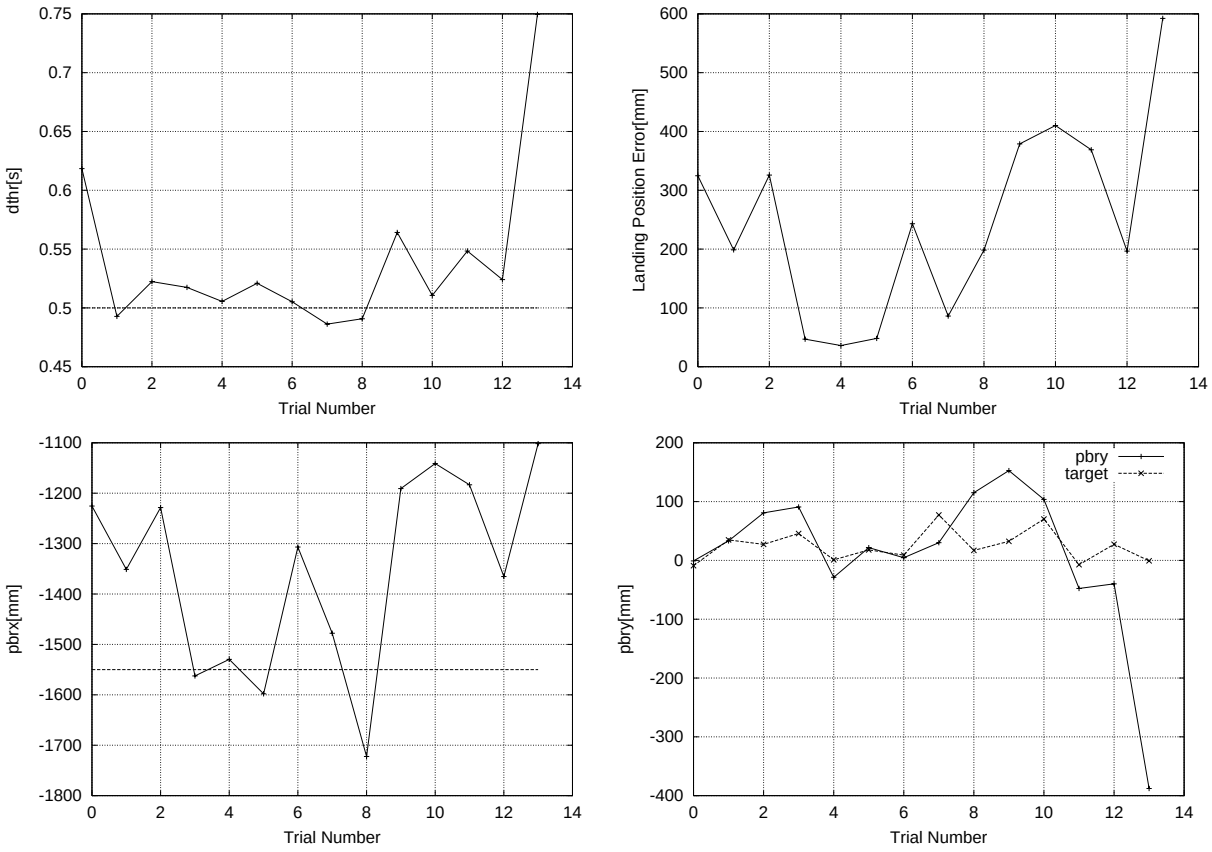


図 5.39　Duration of Flight and Landing Position

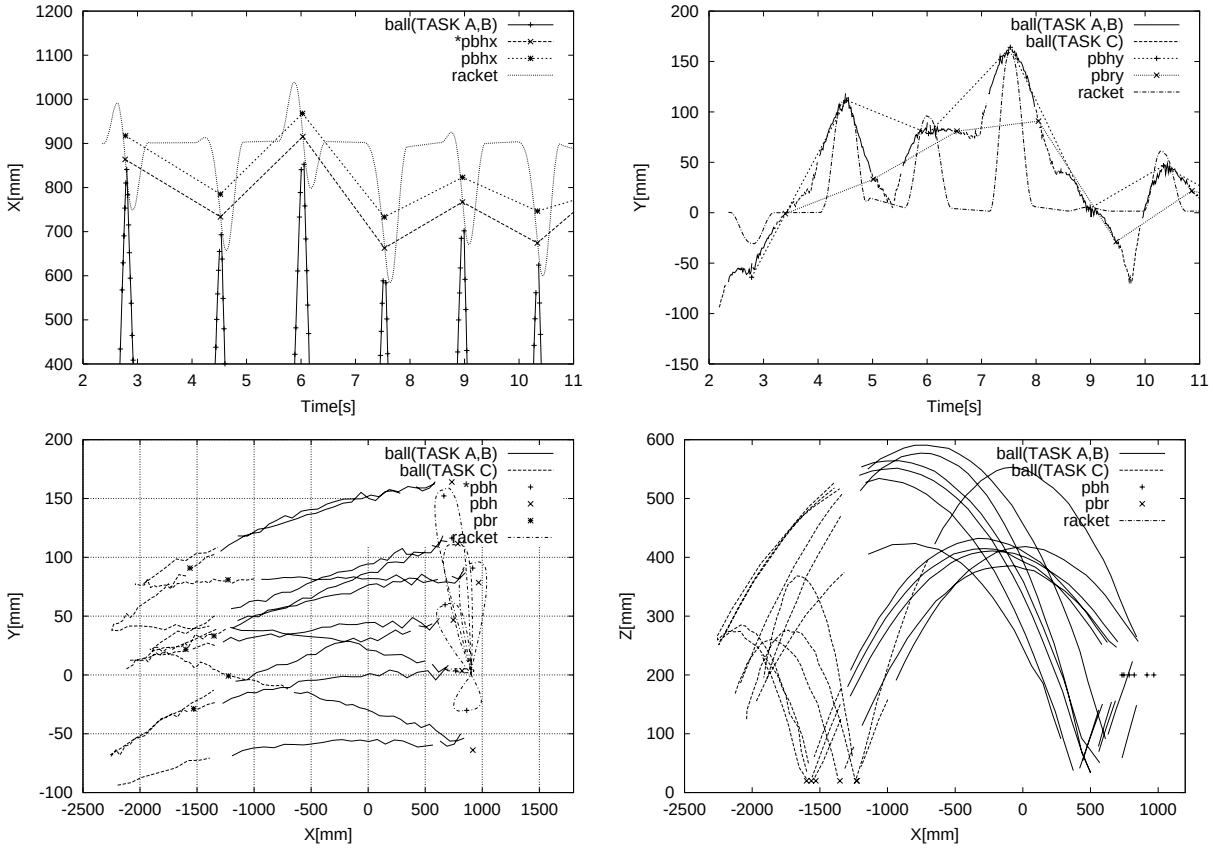


図 5.40 Ball & Racket Trajectories(Before 11[s])

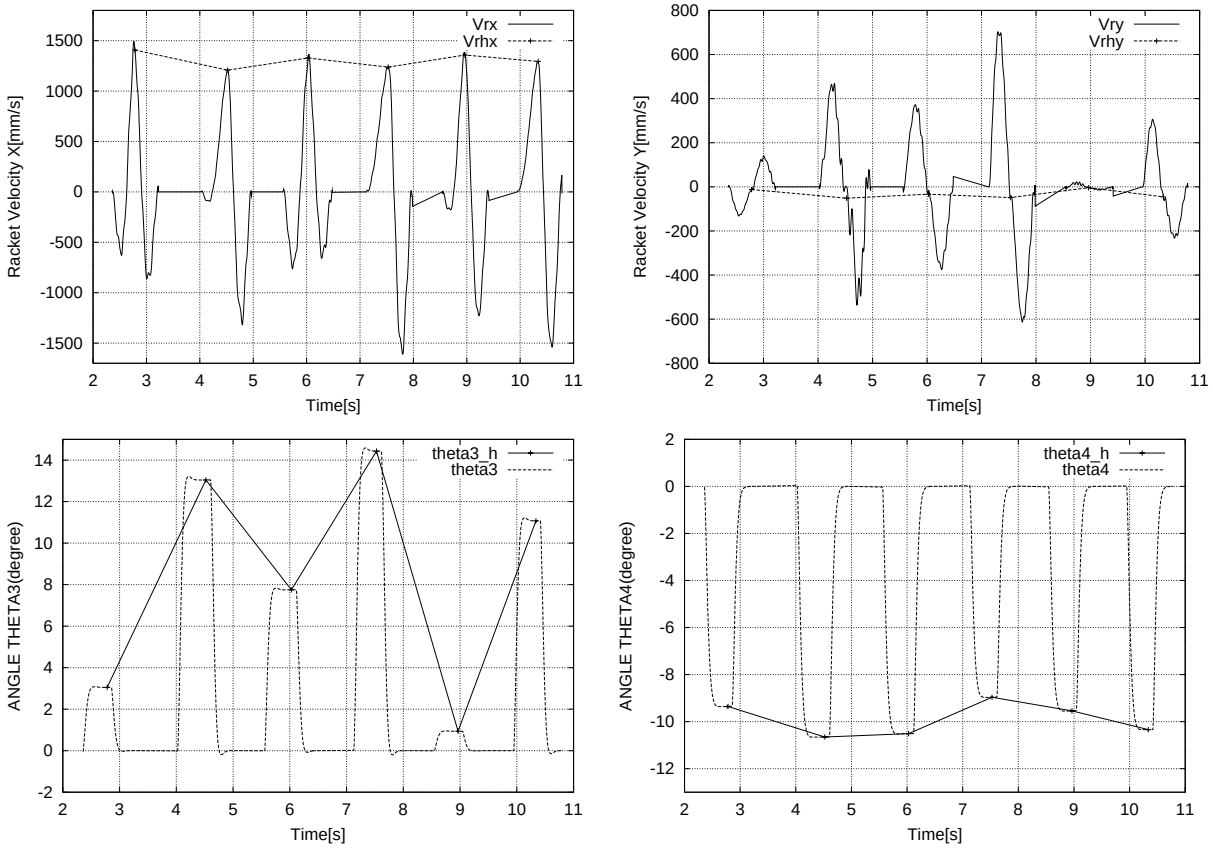


図 5.41 Racket States (Before 11[s])

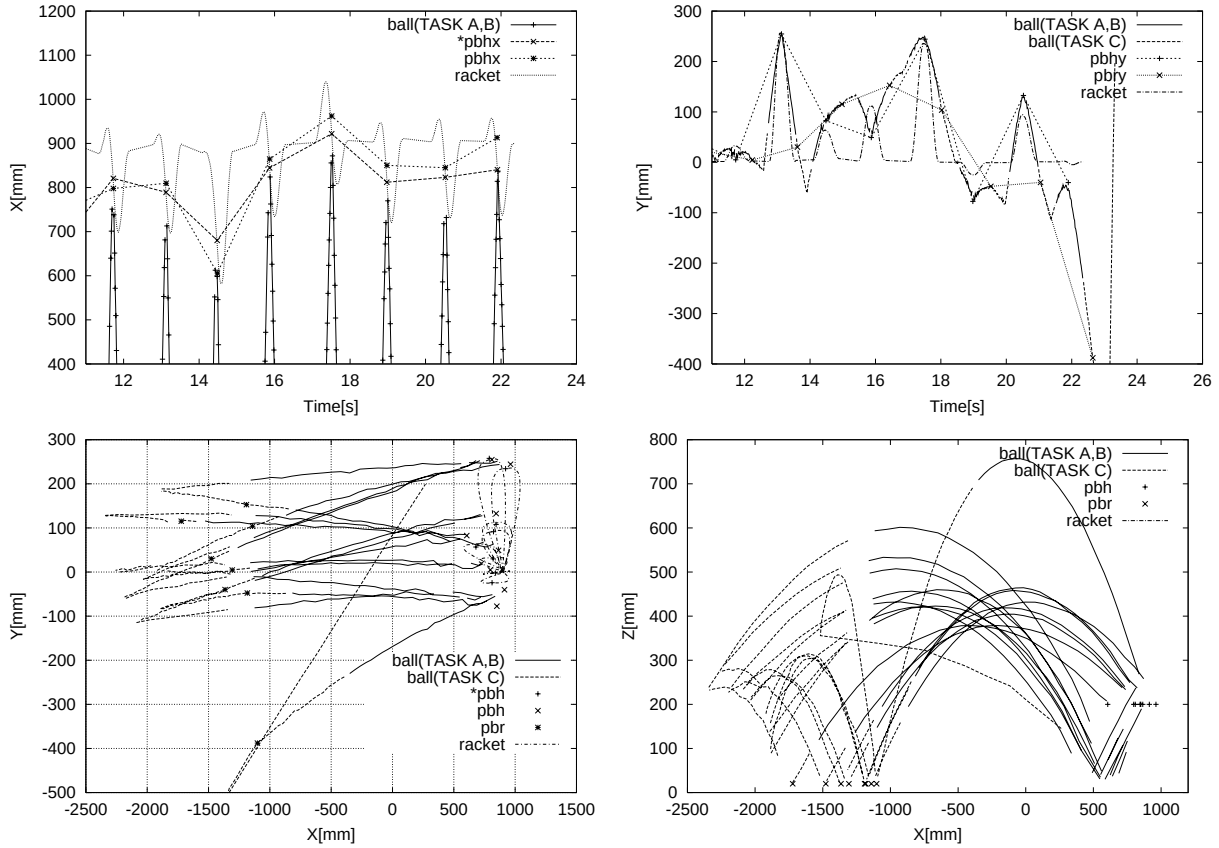


図 5.42　Ball & Racket Trajectories(After 11[s])

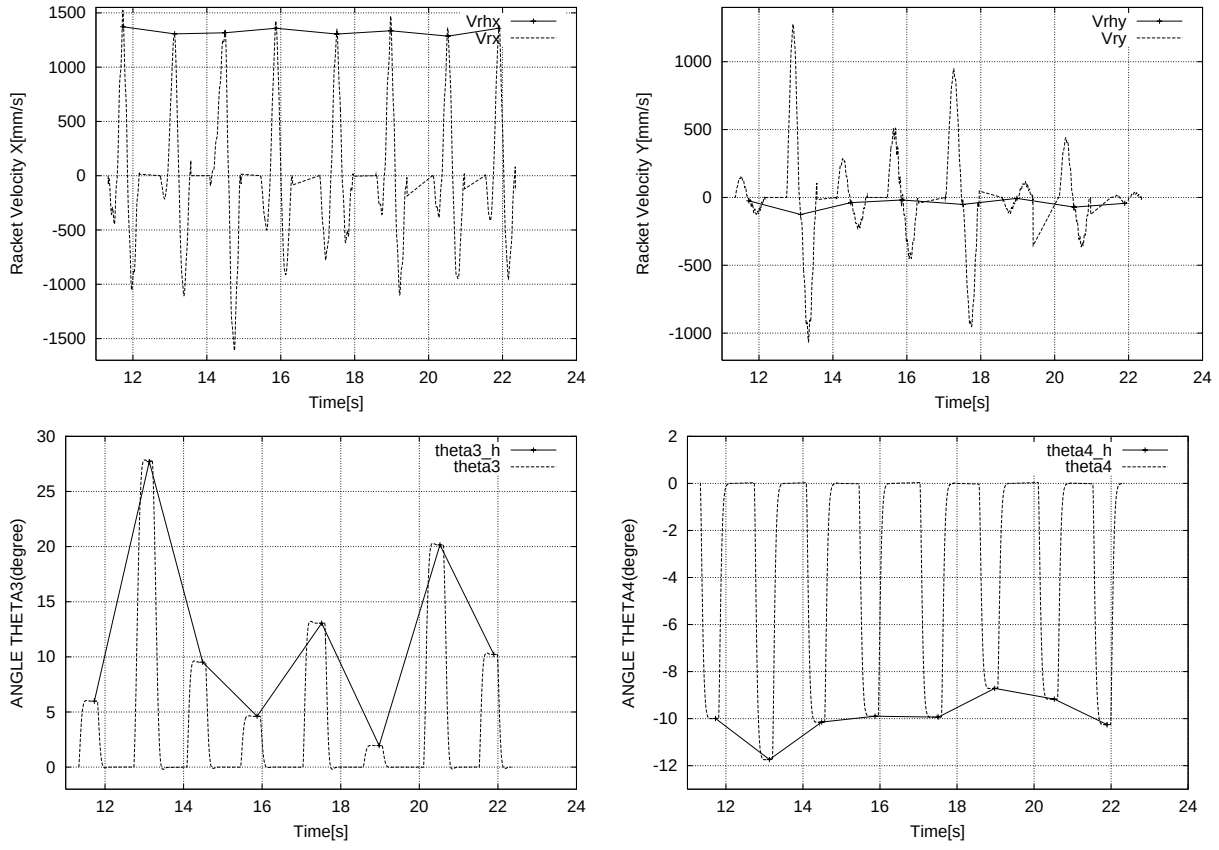


図 5.43　Racket Statuses (After 11[s])

5.8 まとめ

本章では，本論文で提案した手法を用いて対人ラリーが行える事を示した．

本論文で用いたロボットは，可動範囲が狭く，加減速に時間を要することから，打撃動作のための時間，空間を確保が難しい．次に，仮想ターゲットの考え方とミラー理論を応用して実現した卓球タスクは必要最小限のラケット自由度と制御パラメータを用いて行ったものであり，打ち返し可能なボールの速度やコースが限定されていたため，より柔軟にボールを返球できるシステムを開発した．まず，卓球タスクにおける 3 つの物理現象を入出力マップによって表現し，マップ 1 による打撃時のボール状態を予測し，マップ 2，マップ 3 の逆マップを利用して目標の飛行距離と飛行時間を実現するためのラケット速度およびラケット角度を仮想ターゲットとした．マップの実装は LWR によって行った．LWR マップを用いて作成した動作計画を Direct ILC によって正確に実現することで，ボールの飛距離と飛行時間を制御して返球するボール操作タスクを実現した．

また，ロボットが人間にとって打ちやすいボールを，人間がロボットに打ちやすいボールを打つことで，ラリーの継続をめざした．このような限られた条件でのラリータスクではあるが，ロボットは飛来するボールの状態と目標から打撃時のラケット状態を決定し，そのラケット状態を正確に実現する打撃動作を生成することで適切なボールの返球を可能とした．また，1 行程平均約 1.5[s] という短時間の打ち返し動作を繰り返し行うことにより，対人のラリーを継続させることに成功した．図で示した通り，人間の打ち出すボール軌道は毎回異なっているが，ロボットは相手コート目標位置付近にボールを打ち返すことに成功している．これは単一動作では成し得ないことであり，ロボットがロボット自身の動作も含めた卓球タスクにおける現象を学習することにより実現したと言える．

ロボットも人間も，目標通りの打撃に失敗することがあるが，お互いが相手に打ちやすいボールを打ち返すことで，その失敗を補正していく様子が見てとれた．また，ラリー時の人間の打つボールとロボットの打つボール軌道の違いも興味深い．ロボットは，ラケット中心でボールを捕らえると計画しているため，打撃時刻，位置に選択の余地がないが，人間はロボットの打ったボールをロボットの打ちやすいボールへと打ち返すために，打撃高さもボール軌道に応じて選択している様子がボール軌跡から見てとれる．同じ高さで姿勢と速度を調節してボールを打撃し，ボール軌道を操るロボットも面白いが，打撃時刻，位置を選択する柔軟さが人間の特徴であり，幅広いボールに対応できる要因だと想像できる．こういった人間の柔軟さをロボットに導入することは今後の課題である．

第 6 章　人間の打撃動作の計測

6.1　はじめに

人がスポーツを始める際，まずは基本的なフォームを身につける．卓球においても同様である．これは，打撃動作はある一定のパターンによって行われるためと考えられる．ある特定のパターンのボールが来た場合には確実に返球出来るような基本フォームを身につけておき，新たなボールに対して，その基本のボールパターンとの相違を考慮して適切に打撃を調整すると考えられる．人の動作をもとに動作の基本パターンを作成し，アトラクタの形を用いて運動の類似性を認識することによって，模倣学習を行う研究がなされている [15]．他にも，模倣を取り上げた研究は数多く存在するが，それらのほとんどは空間的な運動の模倣であり，時間には考慮していない [16, 17]．我々は運動そのものの模倣学習を一段進めて，外界の環境変化に対する対応能力の学習というところに焦点を当てて研究する．

運動を記述する方程式を環境変化を表す対象物体 (卓球タスクにおけるボール) の運動を制御対象となるロボットの運動にマッピングすることを目的として，手先や身体の位置および速度情報から，ボールの運動に対する人の動作パターンの傾向を取得し，学習制御を行うための単位に分解する．その中で，ボールの運動と関連性のある動作とそうでない動作を，相関値を用いて分類し，基本動作と環境変化への対応動作を別々に扱うことによって，対応能力の向上を目指す．

人間の動作をロボットの制御に適用するために，人間の動作の計測を行い，人間の計測データを利用した予測や行動決定を行う手法を考察する．

6.2　人のスイングの計測実験

人間の打撃動作の計測は，2.2 節で紹介したシステムを用いて行った．打撃動作を計測するにあたり，関係する要素は次のように数多くある．例えば，ボールについては，ボールを打ち出す打球機の位置，角度，打ち出す速度，回転，落下位置，軌道，返球するスイングの種類 (フォアハンド，バックショート，カット，ドライブなど)，返球する目標落下位置，返球の速度，軌道などである．しかし，解析を行うことを前提に計測しなくてはならないので，以下のように条件を絞って計測を行った．

- 打球機の位置は固定．
- 目標バウンド位置を指定する．
- 供給するボール速度および角度はバウンド位置とボールの速さによって決定．速さは “高”，“中”，“低” の 3 種類を用意した．

- 供給するボールの回転はナックル.
- 打撃方法は，フォアハンド (右利きなので身体の右側で打撃する)
- 返球の速度は (“速い”，“遅い”) を指示する場合と “指示しない” 場合がある.
- 返球も目標落下位置を指定する.

以上の条件を前提に，以下の場合について共通点と相異点を調べた.

① 異なる目標返球位置に対する打撃動作

② 異なる供給ボール速度に対する打撃動作

③ 異なる返球ボール速度に対する打撃動作

④ 異なる供給ボールバウンド位置に対する打撃動作

6.3　実験 1:目標返球位置の変化

6.3.1　実験条件 1

まず，打球機からのボールを固定し，目標返球位置を変化させた場合の打撃動作の相違について調べた．この実験は，x 方向 (ネットに垂直な方向)+300mm および + 700mm かつ y 方向 (ネットと平行な方向)+300mm および − 300mm の 4 箇所に+印を付け，人間側のコートの x 印付近 $(1000, 0)$ でバウンドするように打球機で送球した．人は打球機側コートの印付近を目標に返球した (**図 6.1**)．打球機からのボールの落下位置はおよそ (1000, 0) の位置になるようにした．1 つの目標につき，それぞれ 300 球のボールに対する打撃動作を計測した．ボールは 2 秒おきに打ち出し，100 球ごとに止めてデータを保存した．得られたボールの位置データは，x, y 方向は 1 次式，z 方向は 2 次式により，7 点づつを用いて最小二乗法近似を行い，速度および加速度を計算した．また，FASTRAK によるラケットおよび肘の位置データに関してもこちらは 5 点づつを 2 次式で近似して速度，加速度を算出した.

700mm の位置を目標に打ったものを long，300mm の位置を目標に打ったものを short とし，$y = 300mm$ の目標位置を left，$y = -300mm$ の目標位置を right とする.

6.3.2　実験結果 1-1:落下位置

上記の計測実験の結果について，まずはボールの落下位置について調べ，人が目標位置への打ち返しをどの程度実現できているかを調べた．ボールの落下位置は，打撃後の点から台上で反発したと思われる点の 2 点前までの座標を最小二乗近似した式から $z = 20$[mm](ボール半径) となる位置を求めた．落下位置の誤差分布を，目標位置ごとに**図 6.2** ∼ **図 6.5** に示す．全てのプロットに重ねて，誤差 dx,dy が $\pm 50mm, \pm 100mm$ 以内のものを矩形で囲んだ.

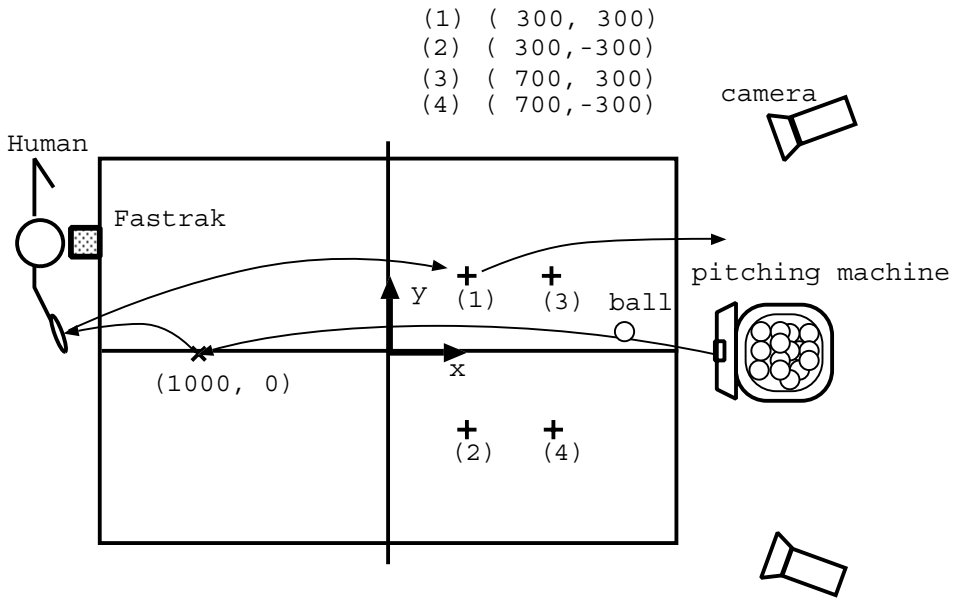


図 6.1 人間の打撃動作計測 (実験条件 1)

また，X および Y 座標が $\pm 50mm, \pm 100mm$ の位置に落下したボールの割合および，落下位置の平均と標準偏差を**表 6.1** に示した．$\pm 50mm$ 以内に落ちたボールは約 20%，$\pm 100mm$ 以内では 60%前後となる．

6.3.3 実験結果 1-2:バックスイング

次に，打撃開始点 (バックスイングが終了し前方に動作を開始する点) について調べた．バックスイングが終了する位置は前に向かって打撃スイングを始める位置であり，スイングの要素の中でも重要なものであると考える．**図 6.6** に，実験中に行った打撃の内，打撃したボールが狙いどおり目標位置付近 (半径 7.5cm 以内) に落下した打撃のバックスイング終了位置とそのスイングの平均軌道を示す．また，打撃時刻を t=0 としたときのバックスイング終了時刻の分布をヒストグラムとして**図 6.7** に示す．

バックスイング終了位置は，目標によって明らかに傾向が異なる．short の場合は left, right 共に同様の分布を示しており，long の場合は，left の方が前方で小さなバックスイングで打撃を開始し，right の方が後方，特に y 方向に大きくバックスイングを取ってから打撃している．全体としてはひとつの円軌道の上に乗っているように見える．また，long へ打撃する時の方が x 方向に大きくバックスイングを取っている．遠くに打つ場合，打撃方向の速度を得るために大きなバックスイングを取っていると考えられる．y 方向に関しては，打撃時のラケットの速度方向に関係があると考えられる．打撃時のラケットの速度方向については次節にて述べる．

打撃開始時刻は，全体を見ても，目標ごとに見ても打撃時刻の約 0.15 秒前を中心に開始

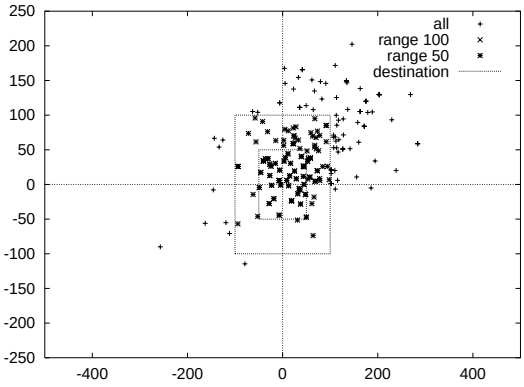


図 6.2 落下位置の誤差 dx-dy(目標 (300, 300))

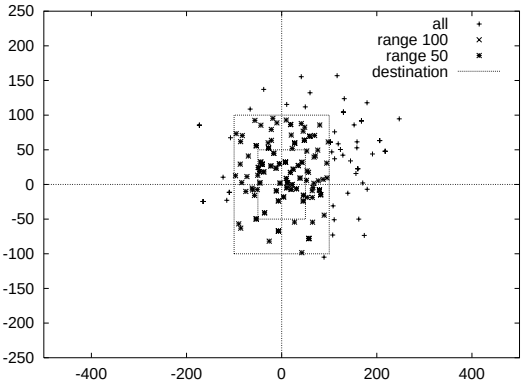


図 6.3 落下位置の誤差 dx-dy(目標 (300,-300))

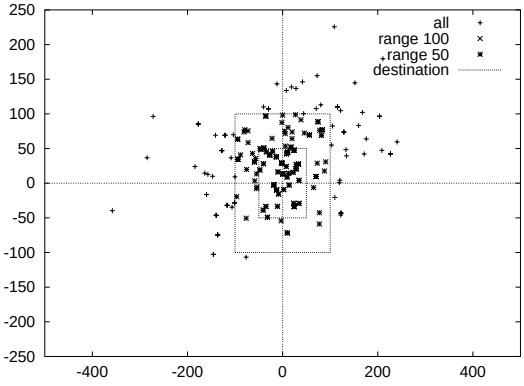


図 6.4 落下位置の誤差 dx-dy(目標 (700, 300))

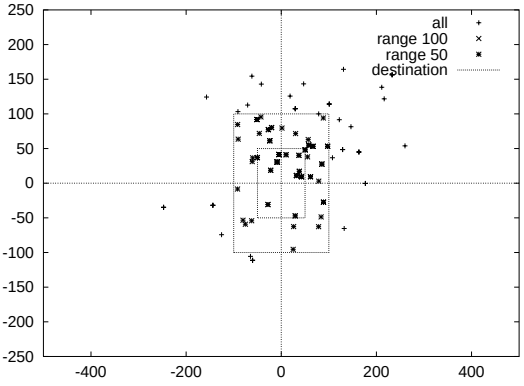
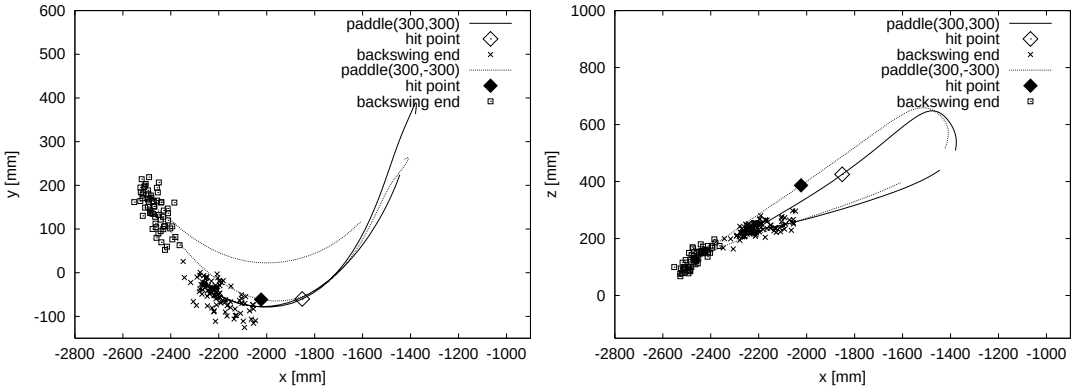


図 6.5 落下位置の誤差 dx-dy(目標 (700,-300))

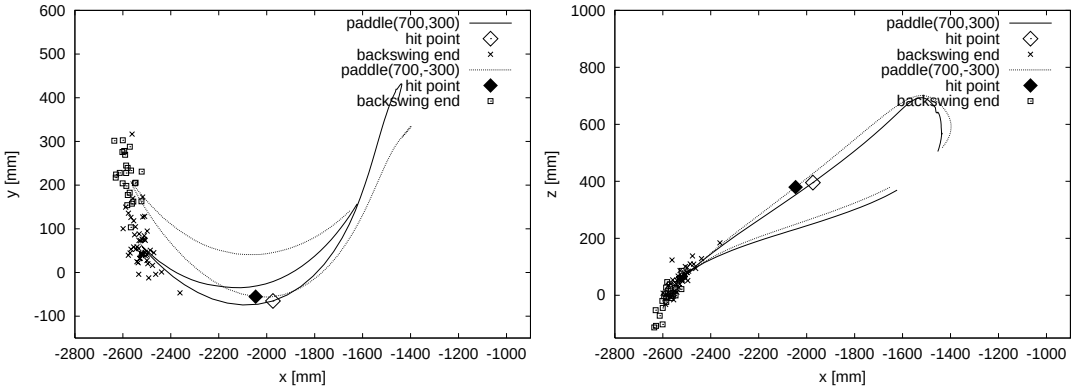
している．また，全体では 0.3 秒前から 0.1 秒前に分布しており，short の方が long よりもバラツキが多少小さい．long-right は絶対数が少ないが，もっともばらついている．しかしながら，バックスイングの位置が目標ごとに大きく異なるにも関わらず，そのタイミングはどの目標へ打ち返す場合も同じだということは，非常に興味深い．

6.3.4　実験結果 1-3:打撃時のラケットの速度方向

フォワードスイングの平均軌跡に「打撃位置と目標位置を結んだ直線」を重ねたものを**図 6.12** に示す．打撃位置から目標位置への方向と打撃位置におけるスイング軌道の接線がほぼ等しくなっていることが見て取れる．つまり，ラケットの x-y 平面での速度方向は目標位置の方向に近くなっていると言える．前節で，右側の目標位置に打撃する場合にバックスイングの y 座標が大きくなっていたのは，打撃時のラケットの速度方向を右側へ向け



目標位置 (300,300),(300,-300)



目標位置 (700,300),(700,-300)

図 6.6 目標位置ごとの平均スイングとバックスイングの終了位置

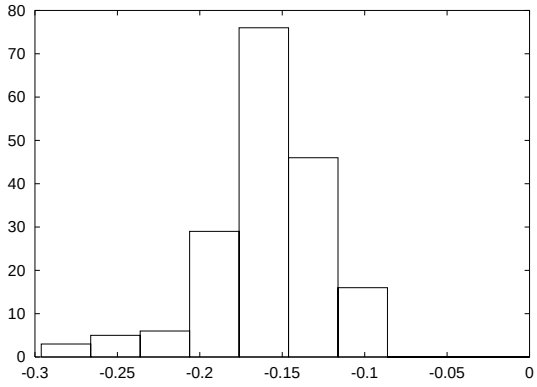


図 6.7 打撃開始時刻のヒストグラム (all)

表 6.1　落下位置の目標達成度

destination [mm]	range [mm]	number/total [times]	rate [%]
(300, 300)	± 50	64/301	21.26
(300,-300)	± 50	69/301	22.92
(700, 300)	± 50	52/282	18.44
(700,-300)	± 50	40/243	16.46
(300, 300)	± 100	179/301	59.47
(300,-300)	± 100	173/301	57.48
(700, 300)	± 100	155/282	54.96
(700,-300)	± 100	117/243	48.15

表 6.2　全体の平均値と標準偏差

destination	average x	average y	stddev x	stddev y
(300, 300)	337.8	328.2	98.47	59.21
(300,-300)	351.4	-298.5	100.6	63.36
(700, 300)	692.4	340.9	111.0	67.10
(700,-300)	744.0	-296.5	119.7	71.44

るためと考えられる．

ただし，目標位置が 右側にあるとき，打撃時のラケット速度方向と目標方向とのずれは大きくなっている．このとき，ラケット速度方向と目標方向が一致する点は，打撃者の身体よりも後ろ側 (x 負方向) になる．この位置で打撃を行うためには，ボールが飛んで来るのを待つか，打撃者が身体を前に移動してやる必要がある．ところが，ボールが飛んで来るのを待つ場合にはボールの高さが現在よりも低くなり，卓球台より低い位置で打撃を行うことになる．また，卓球台が干渉するため身体を前に出すことも不可能である．したがって，方向が一致する地点で打撃を行うことが困難であると判断されるため，身体の前で打撃しているものと考えられる．このとき，ボールの飛ぶ方向はラケットの角度，またはアームの角度によって調整されていると想像されるが，これに関しては正しく解析されるべきである．

6.3.5　実験結果 1-4:スイングの分散

人間のスイング動作の中で，同じ目標位置にボールを運ぶために一定の動作をしている部分があるか，また，あるとすればどのあたりで一定の動作をしているのかを調べるため，目標位置別にラケット軌道の分散を計算した．

目標達成のためのスイングに必要な打撃のタイミング合わせと飛距離調整に大きく関わ

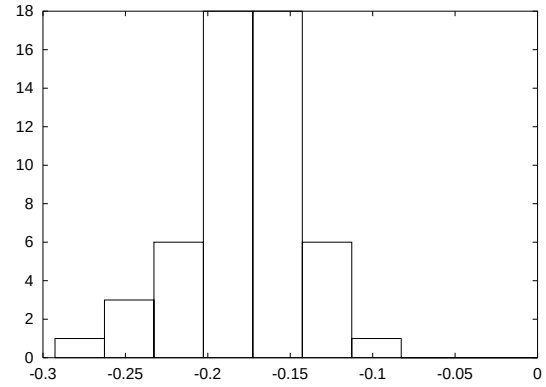


図 6.8 打撃開始時刻のヒストグラム (300,300)

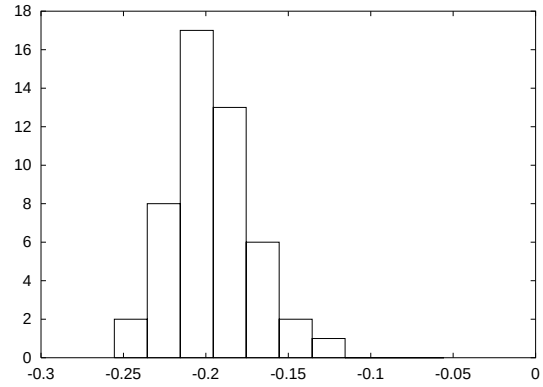


図 6.9 打撃開始時刻のヒストグラム (300,-300)

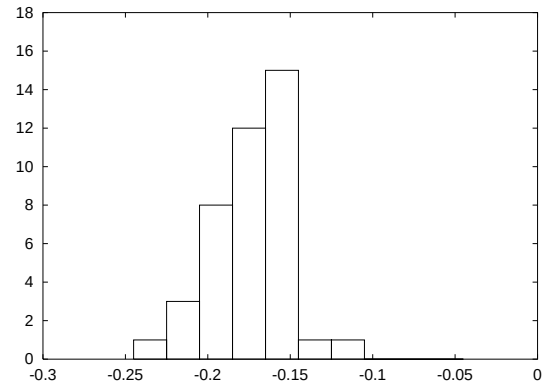


図 6.10 打撃開始時刻のヒストグラム (700,300)

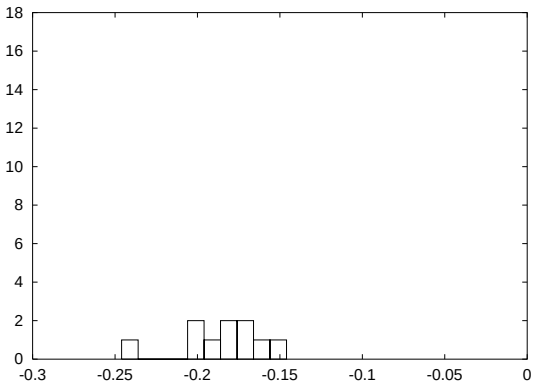


図 6.11 打撃開始時刻のヒストグラム (700,-300)

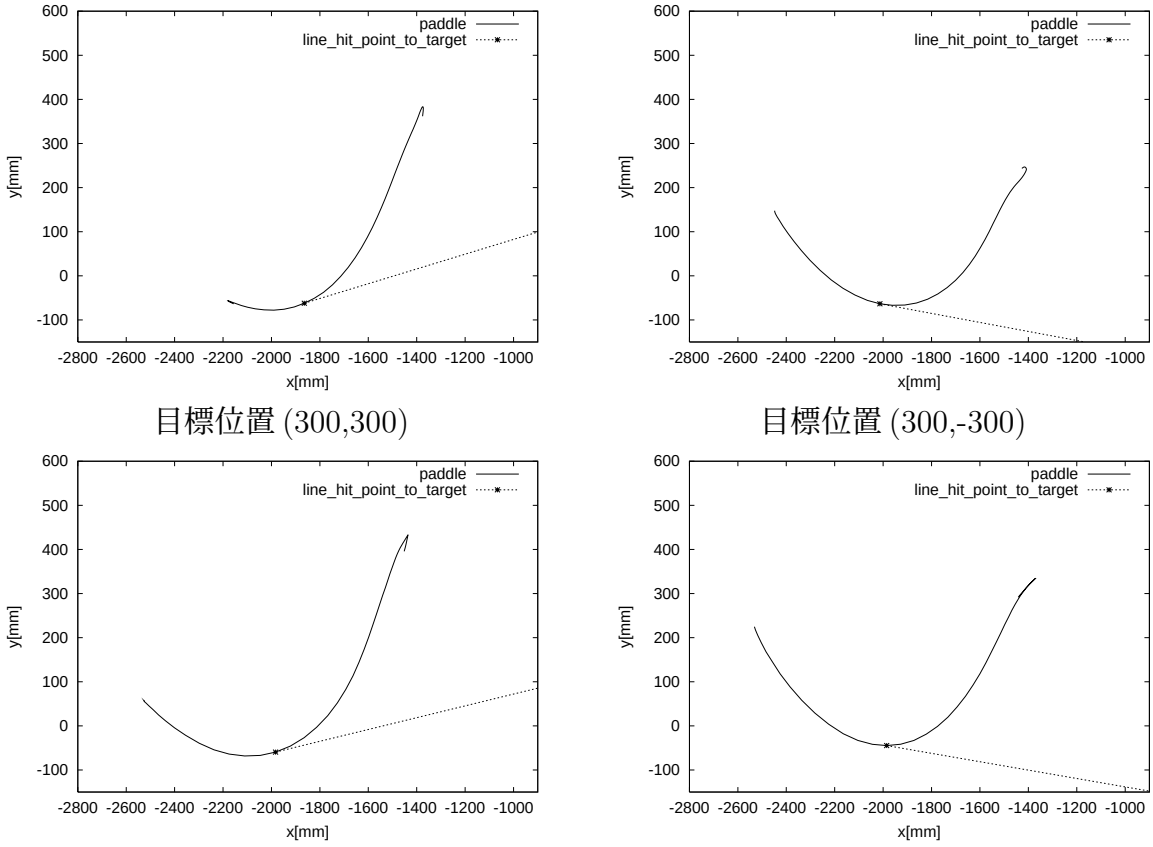
ると考えられるスイング軌道の x 座標のばらつきを目標位置別に**図 6.13** に示す.

バックスイング中にばらつきが大きくなり，バックスイングが完了する辺りで明らかにばらつきが小さくなる傾向が全ての結果に現れている．人間の動きを計測している以上，運動の再現性の限界を考えると，ばらつきが 0 になることはあり得ない．バックスイング完了時に，それ以前に比べて極端にばらつきが減少していることから，バックスイングがおよそ一定の位置で完了していると考えてよいだろう．ただし，**図 6.6** からも分かるように，目標位置が異なると，バックスイング終了位置も異なる．このことから，目標位置に合わせたバックスイング終了位置を決めていると考えられる.

6.3.6 実験結果 1-5:ラケット運動の切替えタイミング

phase の定義

人間の運動の計測を行った研究では，動的な物体に対して動作を行う際に人の動作が予測に基づいた粗い動作と動き出した後に物体に合わせて調整する動作の 2 種類を途中で切替えていることを指摘した文献がある [27].



目標位置 (300,300)　　目標位置 (300,-300)

目標位置 (700,300)　　目標位置 (700,-300)

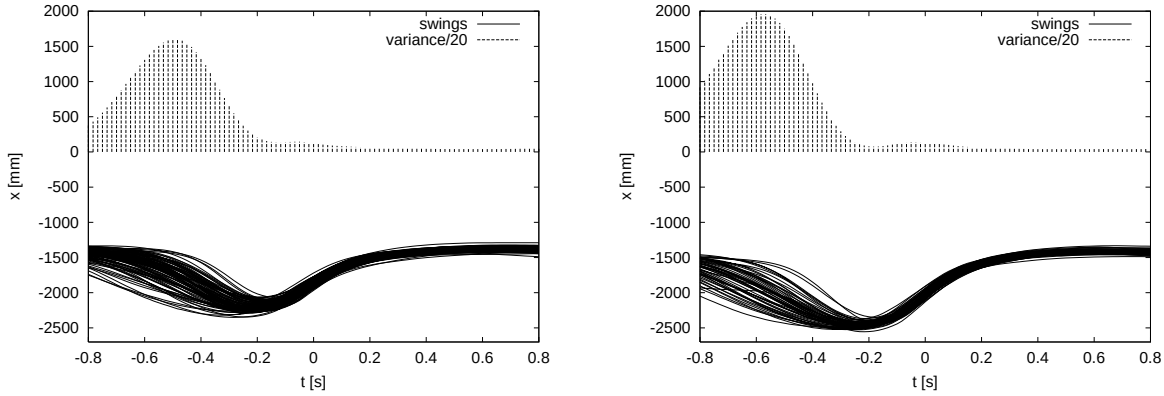
図 6.12　スイングの平均軌跡と，打撃点からの目標位置方向

そこで，ラケットのスイング運動を切替えるタイミングを調べるため，ボールとラケットの運動軌道をそれぞれ**図 6.14** に示すような phase 分けを行った．ラケットに関しては，バックスイング中に加速度が 後向きから前向きに切り替わる瞬間に，ラケットを前向きの運動にするために 力の向きを変えていると考え，バックスイング中に x 方向の加速度が 0 となる所までを phase1 とした．加速度 0 の時点からバックスイングが終了するまでを phase2，バックスイングが前向きのスイングに切り替わってから，再び加速度が 0 となる地点までを phase3 とする．

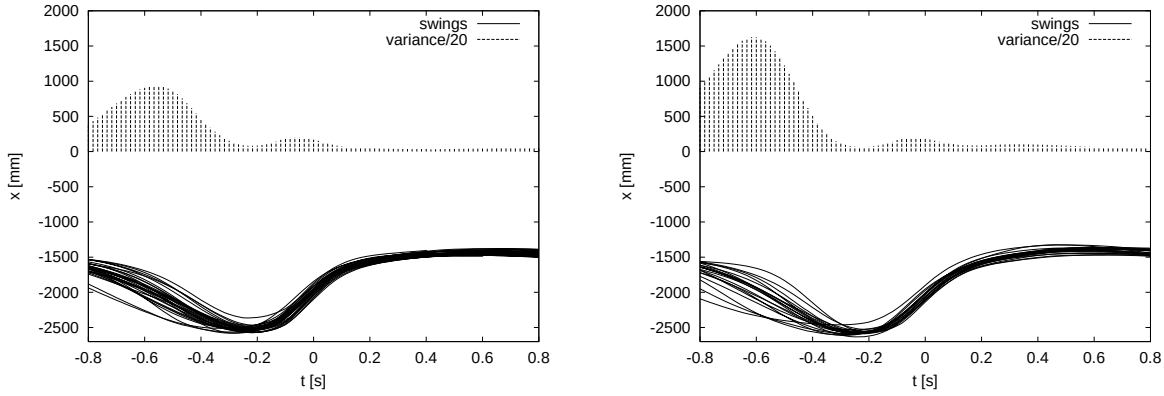
ボールについては，ボールが反発するところを Phase の切替え地点とし，打球機からボールが打ち出されてから始めにコートでバウンドするまでを phase1，バウンドしてからラケットで打ち返すまでを phase2，打ち返してから Quick-MAG 側のコートでバウンドするまでを phase3 とした．

バックスイングと打撃のタイミング

目標地点別のスイングで phase の切替えのタイミングを表したものが**図 6.15** である．それぞれの目標地点について，約 300 回ずつ行った打撃データのうち，目標地点付近 (直径



目標位置 (300,300)　　目標位置 (300,-300)



目標位置 (700,300)　　目標位置 (700,-300)

図 6.13　スイングの x 座標のばらつき

15cm 以内) に返球出来たときのデータのみをプロットしている．飛来するボールがバウンドする直後にバックスイングが終了していること，打撃の前後にラケットの加速度が前向きから後向きに切り替わっていることなどが分かる．打撃の瞬間に加速度が 0 付近になることについては計測の結果と被験者の話から，打撃時になるべく等速に近い状態にすることで，安定した打撃を得ようとしていると考えられる．この結果は，Bootsma が行った世界選手権レベルの卓球選手のスマッシュ動作の傾向 [28] と同様である．

phase1 終了のタイミング

phase1 終了時，つまりバックスイング中に力を後ろ向きから前向きに切替えるタイミングについて，何らかの法則性を期待していたが，目標位置 (300,-300), (700.-300) の結果を見ると，そのばらつきは大きい．ただ，目標位置 (300,300),(700.300) の場合には，phase2 の終了時刻ほどではないものの，ある程度傾向があるように見える．これらの目標位置の場合において，ラケットの phase1 終了時には，ボールがネットを越えたあたりにある場合が多い．目標位置が左側の場合にのみ，phase1 終了時のボール位置に偏りが見られたのは，

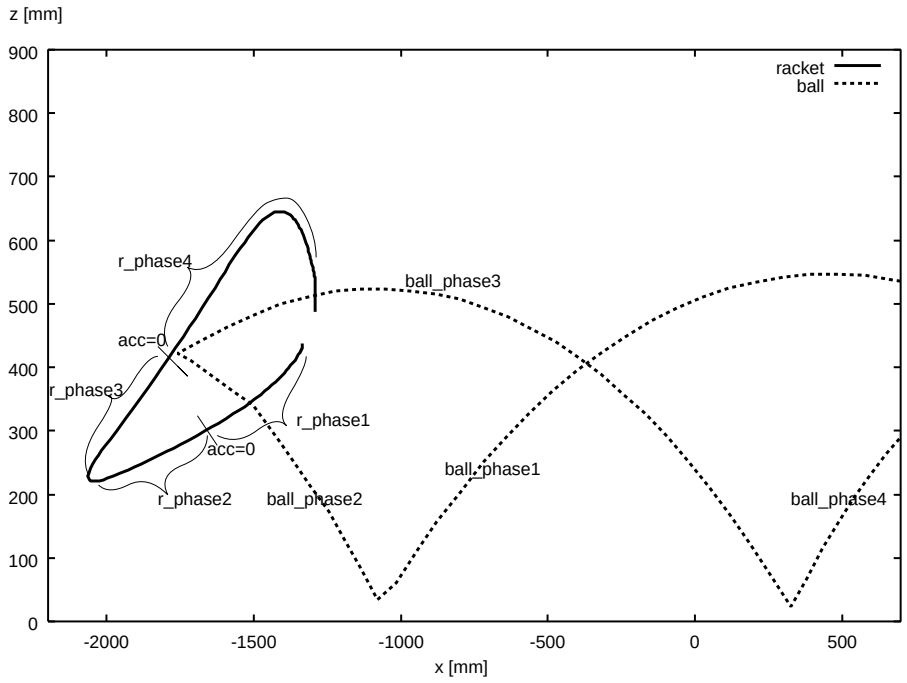


図 6.14　ラケットとボールの phase 定義

左側に打つ場合の制約が少ない (前述のように卓球台の干渉など) ため，打ちやすい左側の目標位置への打撃を繰り返す際に，右側の目標位置の場合に比べてスイングが安定し，タイミングに傾向が出やすくなったと考えられる．

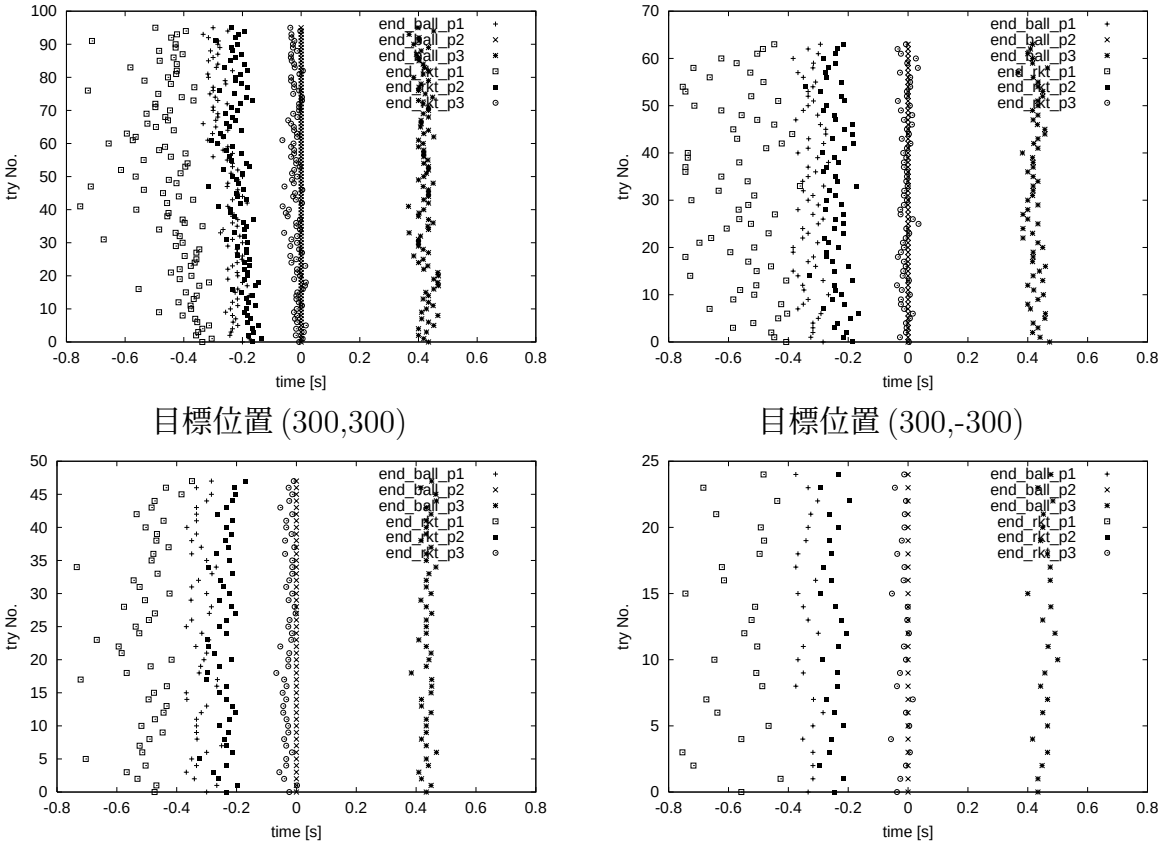
ラケットフェーズ切替え時のボールの位置

ラケットのフェーズ切替え時のボール・ラケットの位置を**図 6.16** に示す．ラケットの phase2 の終了位置は明らかであり，**図 6.15** から phase3 の終了位置はほぼ打撃位置と同じになることが分かっているので，それらの位置は省略し，ラケットの phase1 が終了した時のラケットとボールの位置，および phase2 が終了したときのボールの位置のみを示した．

ラケット・ボールの軌道については無作為に選んだ 1 パターンのみを示している．phase1 の終了位置については打撃時刻を基準にした時刻よりもラケット位置の方に関係深いことが読み取れる．

打撃時間を基準としたバックスイング中のある時間でのラケットの位置はバラツキが大きいことが分かっているが，phase1 の終了位置付近，つまり力の切替えを行う時点から急激にばらつきが小さくなる．**図 6.15** と，**図 6.13** とを見比べると，phase1 が終了するあたりで位置のばらつきが最大になり，そこからバックスイングが終了 (phase2 が終了) するまでの間に急激にばらつきが小さくなっていることが分かる．

以上より，打撃動作を以下の手順で行っていると推測できる．



目標位置 (300,300)　　目標位置 (300,-300)

目標位置 (700,300)　　目標位置 (700,-300)

図 6.15　phase 切替えタイミング (t=0 で打撃)

① ボールが打ち出されてから粗い精度でバックスイング開始 (phase1 開始).

② 一定の位置で力の向きを切替える (phase1→phase2).

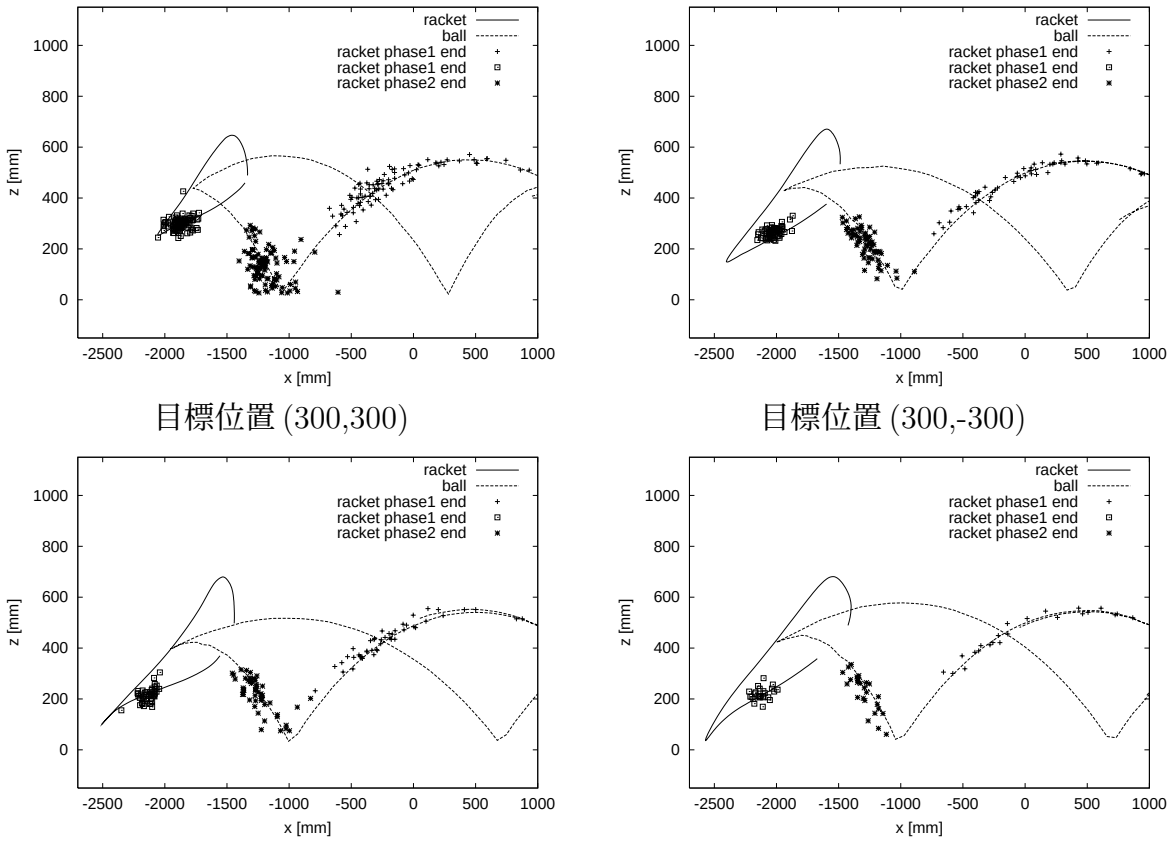
③ ボールとラケットのタイミングを合わせながら，目標位置によって決まる一定の位置でバックスイングを終了 (phase2 終了).

④ 一定の軌道・速度で前向きのスイング開始 (phase3 開始).

⑤ 打撃直前に加速を止めて速度の微調整をして (phase3 終了) 打撃.

6.4　実験 2:供給ボール速度および返球ボール速度の相違

実験 1 では，目標位置の違いによる，スイングの特徴の変化を中心に解析して来た．次は，打ち返す目標位置が固定された場合におけるボールの速度の相違のスイングへの影響を調べた.



目標位置 (300,300)　　目標位置 (300,-300)

目標位置 (700,300)　　目標位置 (700,-300)

図 6.16　phase 切替え時のボール・ラケットの位置

表 6.3　実験内容

表記	供給ボールの落下位置	打球機速度	返球速度
(slow-slow)	-1000mm	遅い (約 $4.2m/s$)	遅い (約 $5m/s$)
(slow-fast)	-1000mm	遅い (約 $4.2m/s$)	速い (約 $7m/s$)
(fast-fast)	-1000mm	速い (約 $7m/s$)	速い (約 $7m/s$)
(fast-slow)	-1000mm	速い (約 $7m/s$)	遅い (約 $5m/s$)

実験環境は 6.3 節での実験と同じであるが，速さの違いによるスイング軌道の変化を調べるために，返球の目標位置を固定し,「打球機から打ち出すボールの速さ」及び「打ち返すボールの速さ」を変化させた場合のスイングを計測した．目標位置としては，前回の実験で被験者が最も打ちやすいと感じた (700,300) を採用した (**図 6.17**)．実験条件を**表 6.3** にまとめた．

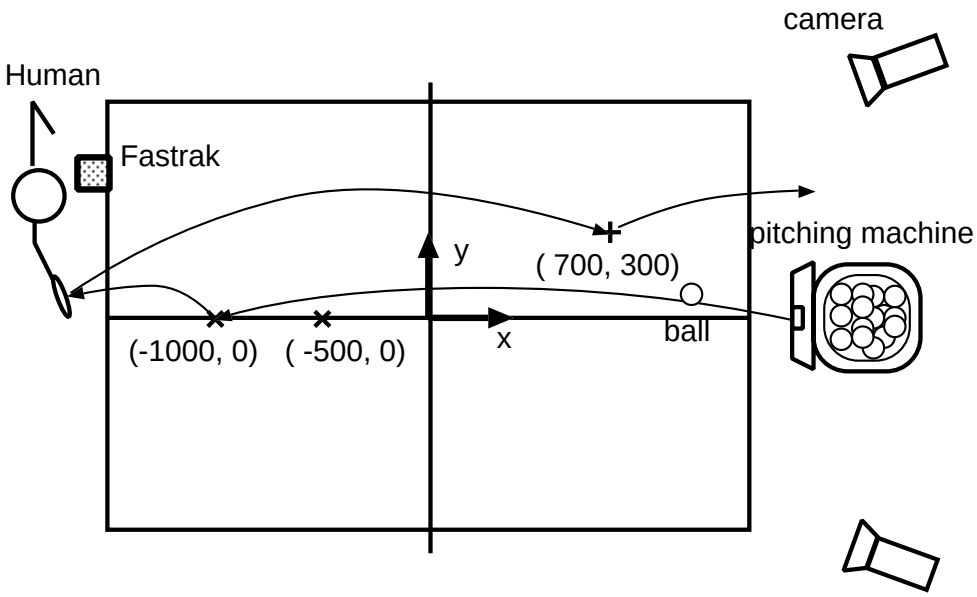


図 6.17 人間の打撃動作計測 (実験条件 2)

6.4.1 実験結果 2:目標の達成度

それぞれの場合について，半径 50mm 以内に落下したものを目標達成とした場合と半径 100mm 以内を目標達成とした場合の目標達成率を**表 6.4** に示す.

また，目標位置との差の平均と標準偏差を**表 6.5** に，ボール落下目標位置からの誤差を**図 6.18** に示した.

y 方向のずれに比べて x 方向のずれの方が若干大きいのは 6.3 節と同じだが，返球が速い時の方がより x 方向にばらつく傾向がある．打球機からのボールが遅い方が達成率が高く，返球と打球機からの供給がどちらも遅い場合が最も達成率が高くなった．興味深いのは，供給されるボール速度が速い場合には，遅く返球するよりも速く返球するほうが達成率が高いことである．このことは，以下の仮説で説明できる．打ち返し目標の達成率が，打球してからボールが目標地点に到達するまでの時間と距離に関係しており，基本的には時間が長いほど，また距離が短いほど達成率が上昇する．ここで，時間が一定より長い場合には時間の方が支配的で，時間が短い場合には距離が支配的になると考えられる.

6.5 実験 3:供給ボールの落下位置の相違による影響

スイングが大きく変わると予想される例として，打球機からのボールが，今までの場合より大きく打球機側でバウンドする場合も計測した．(**表 6.3**) 今までより，早くバウンドすることになり，バックスイングのタイミング等に影響が現れると予想される．打球機からのボールの落下目標位置を (-500,0) に設定し，目標返球位置は (700,300) とした．打球機

表 6.4　落下位置の目標達成度

condition	radius	number/total	rate
打球機：遅, 返球：遅	50mm	24/181	0.133
打球機：遅, 返球：速	50mm	13/120	0.108
打球機：速, 返球：速	50mm	9/125	0.072
打球機：速, 返球：遅	50mm	11/166	0.066
打球機：遅, 返球：遅	100mm	101/181	0.558
打球機：遅, 返球：速	100mm	36/120	0.300
打球機：速, 返球：速	100mm	38/125	0.304
打球機：速, 返球：遅	100mm	52/166	0.313

表 6.5　目標位置との差の平均値と標準偏差 [mm]

condition	diff mean(x,y)	distance mean	diff SD(x,y)
打球機：遅, 返球：遅	(-39.6,27.3)	102.9	(84.1,64.9)
打球機：遅, 返球：速	(11.0,31.7)	179.3	(195.1,72.3)
打球機：速, 返球：速	(30.4,56.7)	146.9	(128.3,79.1)
打球機：速, 返球：遅	(-46.7,44.0)	141.2	(120.0,81.9)

からのボールは速いもの (約 7[m/s]) と遅いもの (約 3.7[m/s]) を用意し，それに対して，打ちやすい速度で返球した.

6.5.1　実験結果 3:目標の達成度

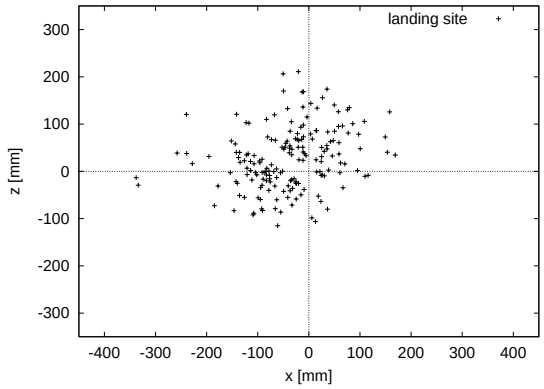
落下位置の目標達成度は，範囲内に落下した割合がそれぞれ,

- 範囲 50mm，打球機：速 → 数: 13/182，割合 7.1[%]
- 範囲 50mm，打球機：遅 → 数: 20/188，割合 10.6[%]
- 範囲 100mm，打球機：速 → 数: 60/182，割合 33.0[%]
- 範囲 100mm，打球機：遅 → 数: 88/188，割合 46.8[%]

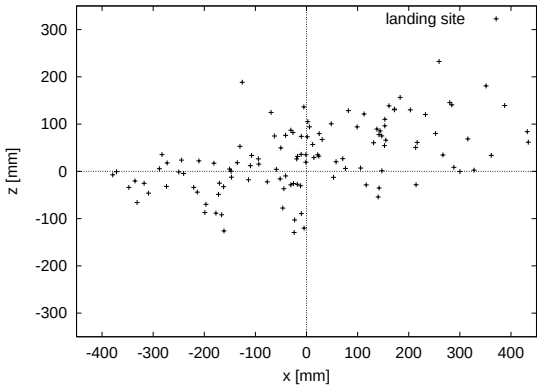
となっており，その位置の，誤差平均，標準偏差，絶対誤差の平均がそれぞれ,

- 打球機:速，平均誤差:(-2.8,36.6)，標準偏差:157.2，距離平均:(158.6,65.7)
- 打球機:遅，平均誤差:(19.3,39.2)，標準偏差:117.7，距離平均:(116.2,51.8)

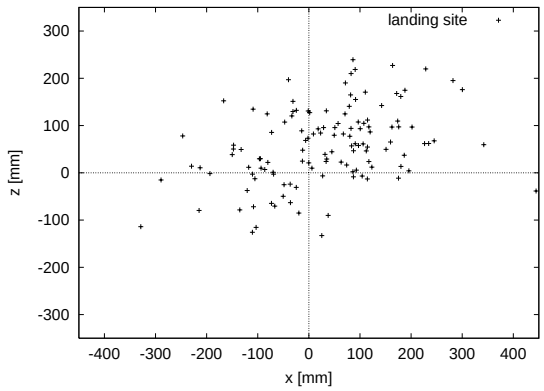
となっている．また，ボール落下位置の目標座標からの誤差を**図 6.19** に示した.



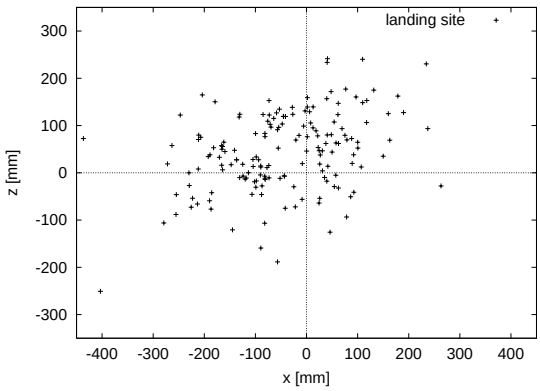
打球機:遅い, 返球:遅い



打球機:遅い, 返球:速い

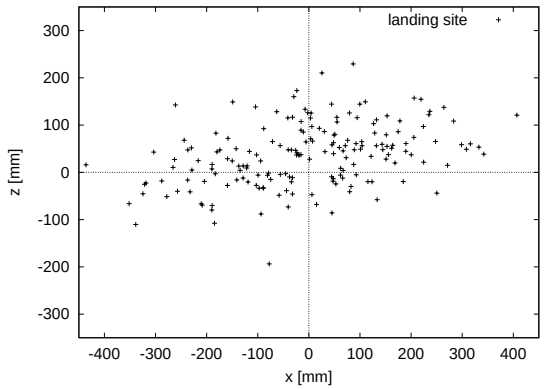


打球機:速い, 返球:速い

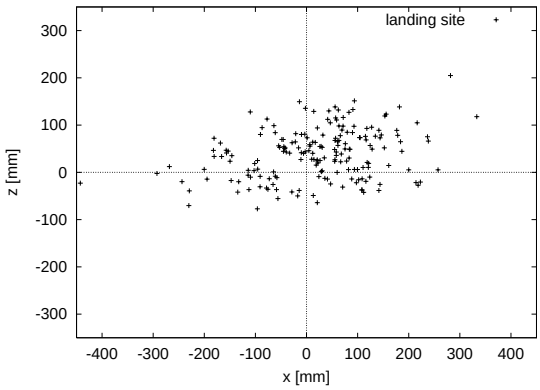


打球機:速い, 返球:遅い

図 6.18 打ち返したボールの落下位置



バウンド:早い, 返球:速い



バウンド:早い, 返球:遅い

図 6.19 打ち返したボールの落下位置

6.6　目標位置ごとのスイングの特徴

実験 1〜3 を総合して，目標位置ごとにスイングの特徴を解析する．

6.6.1　平均軌道とバックスイングの位置

図 6.20 に，実験中に行った打撃の内，打撃したボールが狙いどおり目標位置付近 (半径 75[mm] 以内) に落下した打撃のバックスイングが終了した位置とそのスイングの平均軌道を示す．また，打撃時刻を t=0 としたときのバックスイング終了時刻の分布をヒストグラムとして**図 6.21**-**図 6.28** に示す．打球機の目標位置が $x = -1000$[mm] の場合全てを**図 6.21** に，-500[mm] の場合全てを**図 6.22** に示した．バックスイングが打撃の 0.7 秒以上前に終了することは通常ありえない (ラケット軌道を見ると-0.7 秒以降にバックスイングを開始している)．また，打撃後にバックスイングは行われないので，-0.7[s]〜0[s] 以外のデータはデータ処理時のミスであるとして初めに取り除いた．バックスイング終了 (打撃開始) 時刻は，打球機のボールが長い場合は打撃の約 0.15 秒前が最も多く，短い場合は 0.10〜0.12 秒前ぐらいが最も多い．それぞれの速度について見ると，遅いボールを遅く返すときと短いボールを遅く返すときのみピークが打撃の 0.2 秒前で，他の条件のときには約 0.15 秒前にバックスイングを終了して打撃を行っている．

6.6.2　スイングの分散

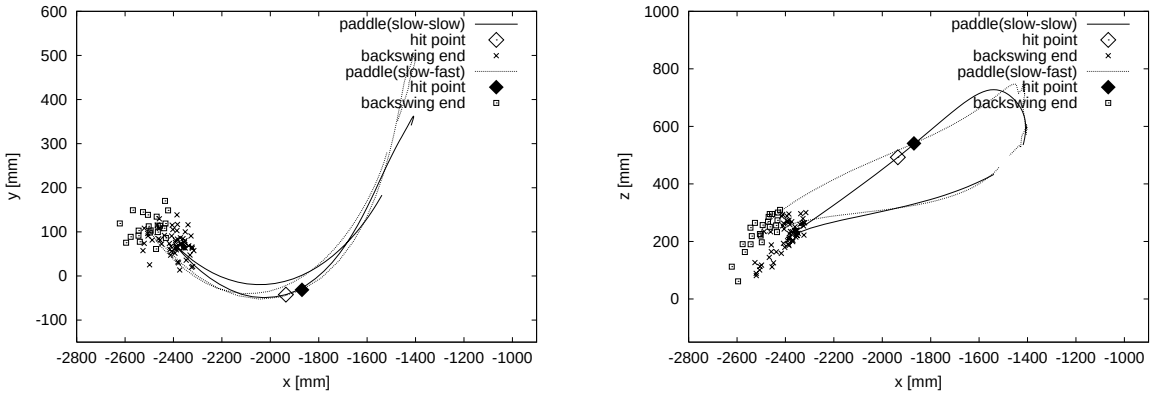
スイングの x 座標の時間ごとのばらつきを**図 6.29** に示す．ここでも，目標位置を変えて計測した時と同じように，バックスイング中にばらつきが大きくなり，バックスイング終了時に，ばらつきが小さくなる．すなわち，飛来する来るボールの速さや，打撃の速さが変わっても，その状況に応じて，一定のバックスイング終了位置が存在すると考えられる．(short-fast)，(short-slow) の場合の打撃終了後にデータ飛びが見られるが，これは他の場合より打球機側でバウンドしたボールに対応するため，より前方で打撃した場合が多く，ラケットが FASTRAK の計測範囲 (**図 2.3**) の外に出てしまったためである．

6.6.3　phase の切替え

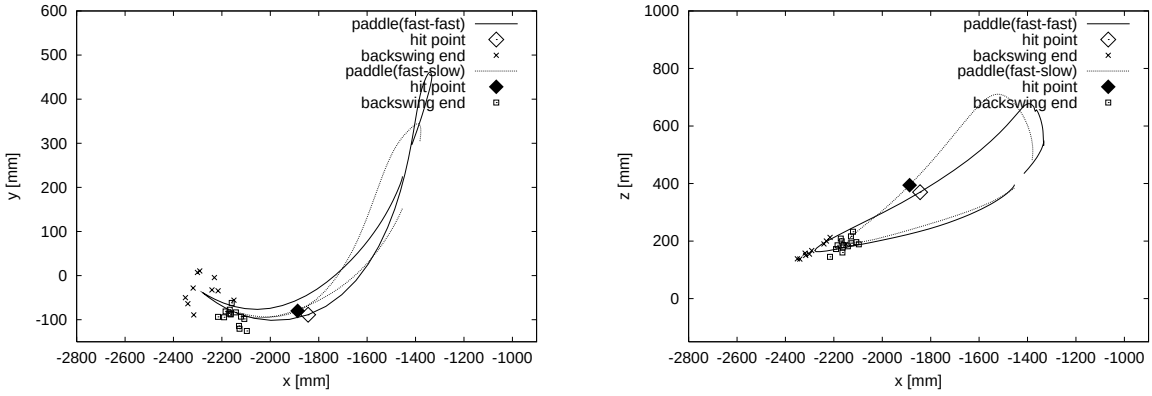
6.3.6 節と同様に phase の切替えについても調べてみた．**図 6.30** に phase 切替えの様子を示す．また，**図 6.31** に，切替え時のボール・ラケットの位置を示す．6.3.6 節節と同様に，ボール供給の速度や打撃速度が変わったときも，その条件ごとに一定の位置で力の切替えをしていることが分かる．

6.7　制御則への応用方法の提案

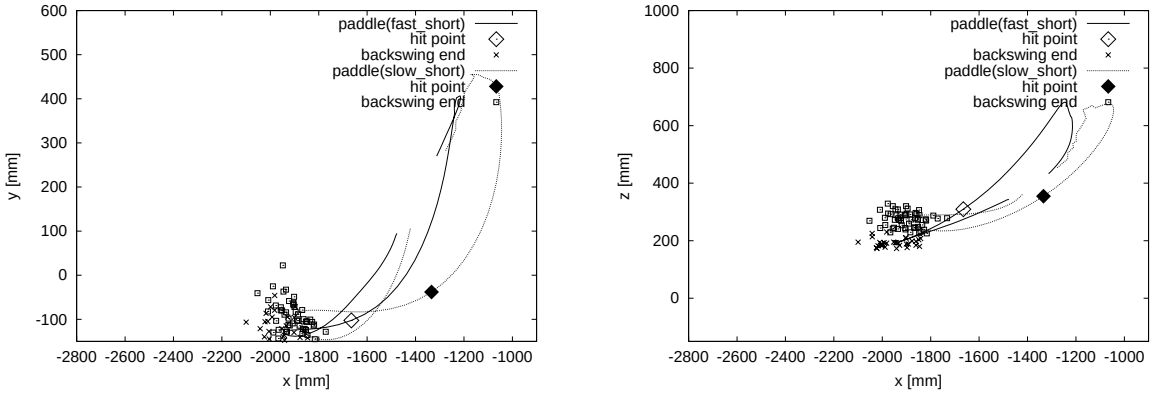
ここでは，6.3 節，6.4 節で述べた条件の違いによるスイングの特徴をまとめ，それらを実際のロボットの制御則に応用する方法を提案する．



(slow-slow),(slow-fast)



(fast-fast),(fast-slow)



(short-fast),(short-slow)

図 6.20 ボールの速さごとの平均スイングとバックスイングの終了位置

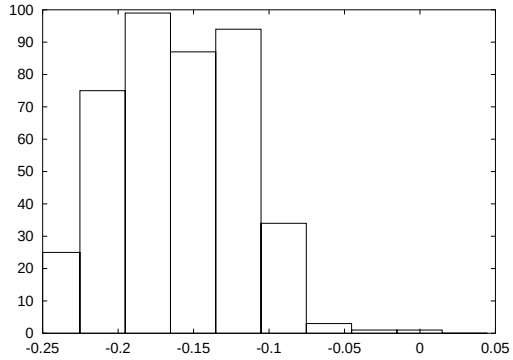


図 6.21 打撃開始時刻のヒストグラム (打球機 long)

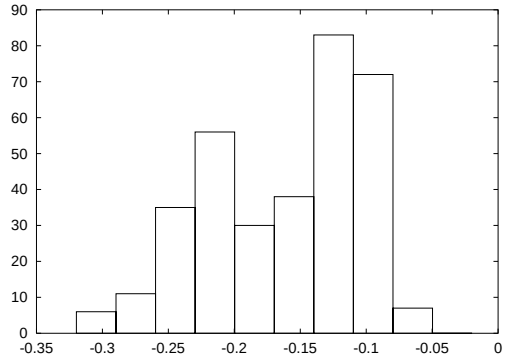


図 6.22 打撃開始時刻のヒストグラム (打球機 short)

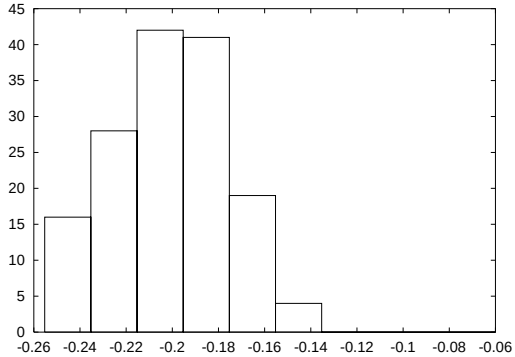


図 6.23 打撃開始時刻のヒストグラム (打球機：遅 返球：遅)

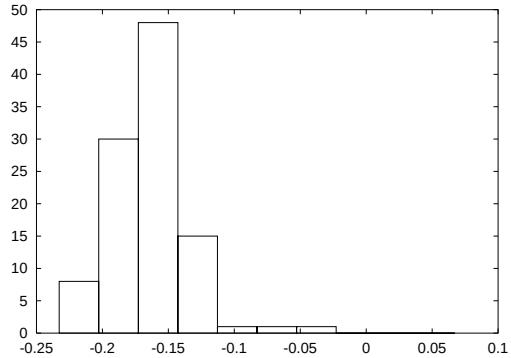


図 6.24 打撃開始時刻のヒストグラム (打球機：遅 返球：速)

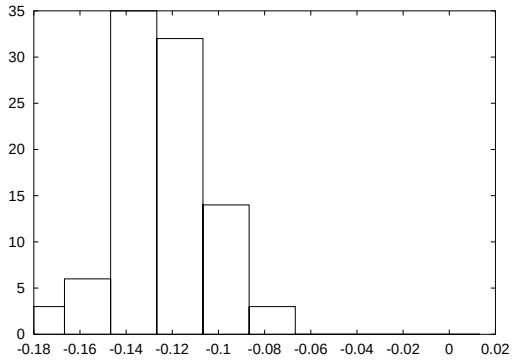


図 6.25 打撃開始時刻のヒストグラム (打球機：速 返球：速)

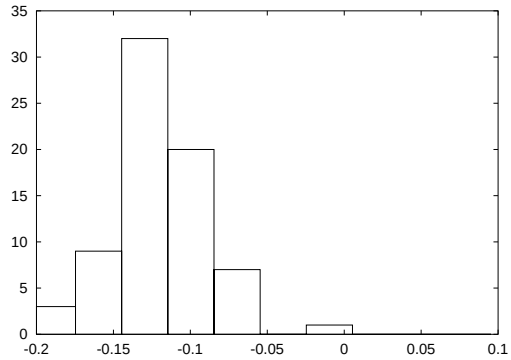


図 6.26 打撃開始時刻のヒストグラム (打球機：速 返球：遅)

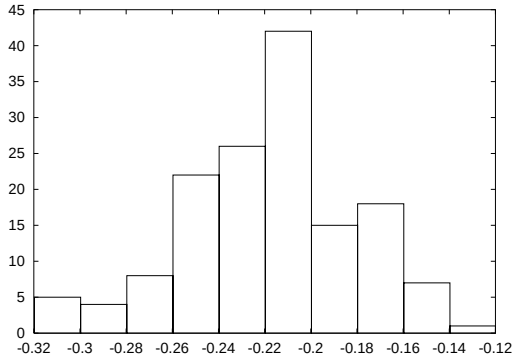


図 6.27 打撃開始時刻のヒストグラム (打球機：短，遅)

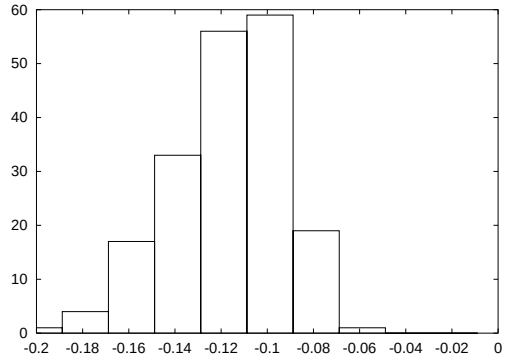


図 6.28 打撃開始時刻のヒストグラム (打球機：短，速)

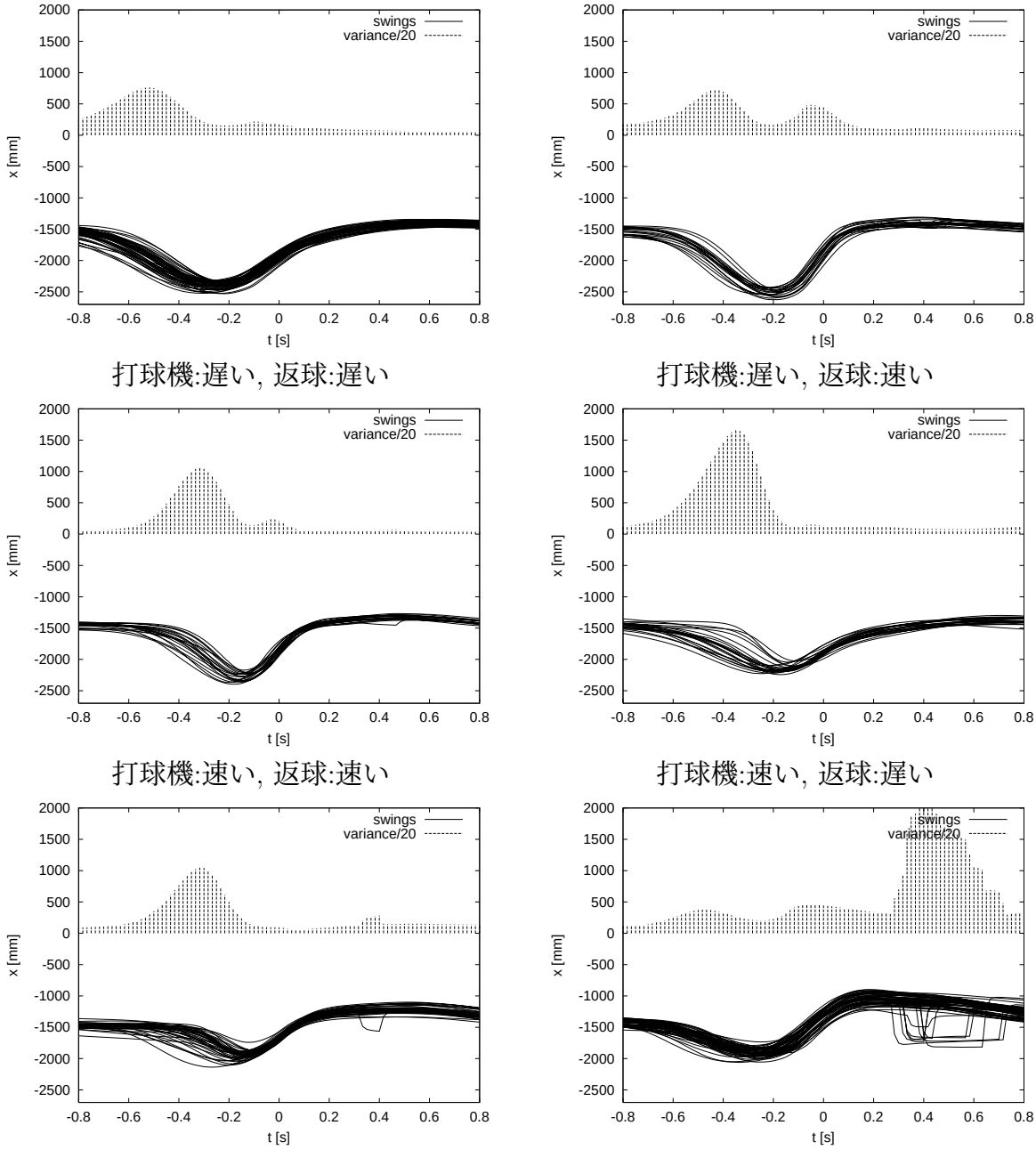


図 6.29 スイングの x 座標のばらつき

6.7.1 スイングの特徴のまとめ

バックスイング終了の位置・タイミングについて

どの場合においても，それぞれの場合ごとにバックスイング終了の位置・タイミングはほぼ一定している．すなわち，飛来するボールの速度・打撃目標位置・打撃目標速度の 3

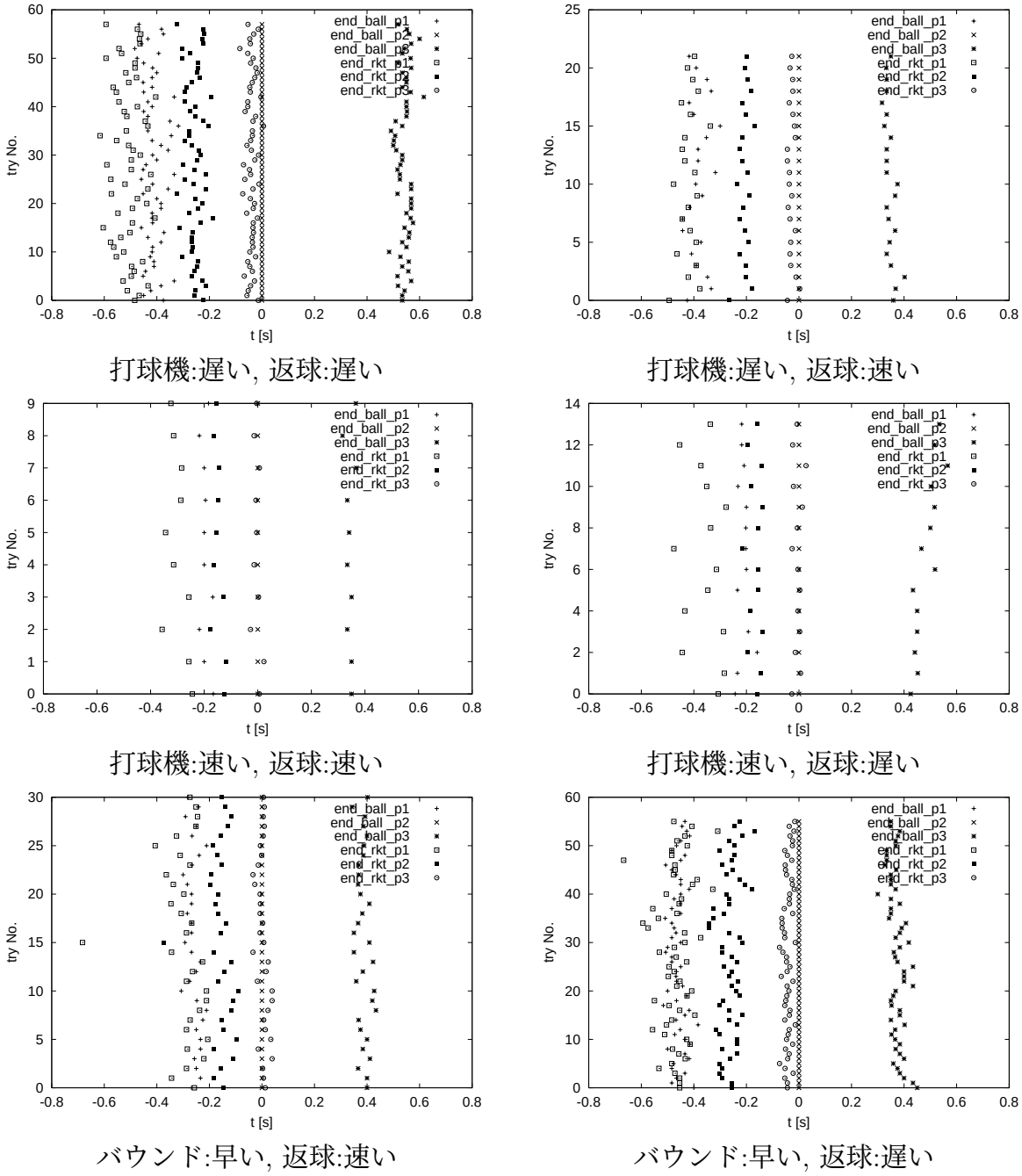


図 6.30　phase 切替えタイミング (t=0 で打撃)

つから，バックスイング終了の位置とタイミングが決定されると考えてよいだろう．さらに，バックスイング終了のタイミングは，どの場合においてもおよそ打撃時刻の 0.2 秒前に集まっている．もちろん，条件の違いによるばらつきは存在し，飛来するボールが速い時の方がバックスイング終了時刻と打撃時刻との差が小さくなる傾向はあるが，どの場合もおよそ同じタイミングでバックスイング終了から打撃にかけての運動を行っていること

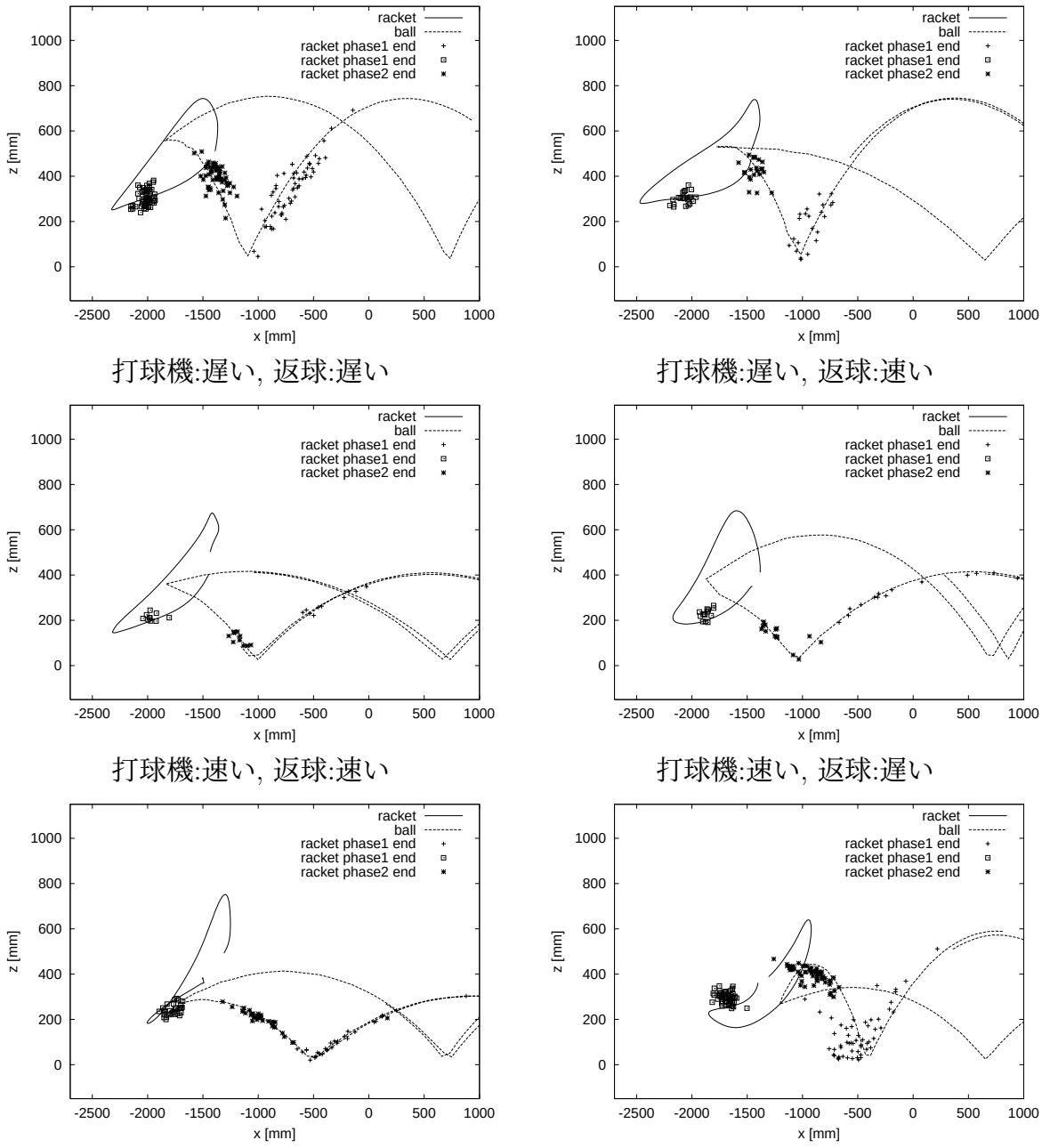


図 6.31 phase 切替え時のボール・ラケットの位置

に注目しておきたい.

打撃の瞬間について

打撃の瞬間については短いボールに対して返球する場合も含めて，全ての場合で加速度が 0 付近になっている．先にも述べたが，これは，打撃時になるべく等速に近い状態にす

ることで，打撃を安定させるため，打撃直前に加速を止めているものと考えられる．

phase1 の終了について

短いボールに対して返球する場合も含めて，それぞれの場合ごとでバックスイング中に運動の切替えを行い加速度が 0 となる場所がほぼ一定している．加速度が 0 となる地点での速度は必ずしも一定ではなく，従って，加速度が 0 となるタイミングも一定になるとは限らない．

スイングのばらつきについて

どの場合でも，バックスイングの開始から phase1 終了までの間に急激にばらつきが大きくなり，phase1 終了からバックスイング終了までの間に再び急激にばらつきが収束する傾向がある．打撃の瞬間のばらつきよりも，バックスイング終了時のばらつきの方が小さい場合が多い．これは，バックスイング終了時には速度が 0 となっているのに対し，打撃時には速度がほぼ最大値となっているため，時間あたりの座標のばらつきも大きくなったものと考えられる．また，バックスイング終了時より若干大きいとはいえ，バックスイング終了時から打撃終了までの間，ばらつきは小さい値を保っている．このことから，バックスイング後は条件ごとにおよそ一定の打撃スイングをしていると考えられる．

6.7.2　人間の打撃の実現方法の仮説

上で述べたように，phase2 が始まるあたりで，スイングのばらつきは最大になり，phase2 が終了するまでに急激に収束している．phase2 が始まるあたりで時間基準の座標のばらつきが最大になるとはいえ，個々のスイングで phase2 が始まる位置は，ほぼ一定している．このことから，ボールが打ち出されてからおおまかにバックスイングを始め，一定の位置で力の向きを切替え (phase2 開始)，そこからバックスイング終了までの間に位置とタイミングを合わせて，前向きの打撃スイングに入ることでボールとラケットとの位置・タイミングを合わせた打撃を実現していると考えられる．

その後，およそ一定の軌道で，打撃スイングを行い，打撃の直前に加速を止め，目標達成のための一定の速度に調整し，打撃を行っている．

6.7.3　ロボットの制御への発展

前述のように，人間の打撃動作から以下の特徴が得られた．

① バックスイングがボールに合わせて大きくばらつくのに対し，フォワードスイングはばらつきが少ない．

② 飛来するボールの速度・打撃目標位置・打撃目標速度の 3 つから，バックスイング終了の位置とタイミングを決定．

③ バックスイング終了から打撃にかけての運動を同じタイミングで行っている

④ 打撃の瞬間については短いボールに対して返球する場合も含めて，全ての場合で加速度が 0 付近になっている．

これをロボットの制御に適用することを考える．さらに，①に示されるように，打撃動作をばらつきのあるバックスイングとばらつきの少ないフォワードスイングに分けて考えると，前半のバックスイングは人が制御してやり，後半は自動的に一定のスイングが行えるようにするという Man-Machine システムが考えられる．しかしながら，最終的に人間のダイナミックマニピュレーションのスキルを完全にロボットに移行してやることが目的であるので，②に示されるような関係で，今まで用いてきた入出力マップを利用してバックスイングの位置とタイミングを予測してやることが考えられる．③，④については，つまりバックスイングさえ決まってしまえば，一定のスイング軌道をとってやることで打撃が行えると解釈できる．

6.8 まとめ

人間のダイナミックマニピュレーション動作をロボットの制御に適用するために，人間の運動や環境変化を表す対象物体 (卓球タスクにおけるボール) の運動を制御対象となるロボットの運動にマッピングすることを目的として，手先や身体の位置および速度を計測した．その中で，ボールの運動と関連性のある動作とそうでない動作を，分類し，基本動作と環境変化への対応動作を別々に扱うことによって，ロボットの対応能力の実現手法の開発を目指した．得られた特徴から，以下のようなロボットのダイナミックマニピュレーションの学習システムを提案した．

あらかじめ人の打撃動作から一定のスイングパターンを生成しておき，ボールに対する調整部分とその一定パターンへの切替えを人が教示してやる．その切替えをマップで求めた位置とタイミングで行うことでロボットが自律的に打撃する．次章より，この方法で打撃を行うシステムについて述べる．

第 7章　マスタースレーブ学習システム

7.1　はじめに

人間の動作をロボットの制御に適用するために，人間の動作の計測を行い，人のデータを用いて予測や行動決定を行う手法を考察した．その結果をもとに，以下のような戦略で人がタスクを実現する際のタイミングや動作パターンおよびボールの挙動を，ボールの速度や位置などの情報とやタイミングや待機位置との関係を表すマップとして直接ロボットが学習しながらダイナミックマニピュレーションタスクを実現する．

卓球の打撃動作パターンは「飛来するボールに対応した打撃動作のための位置調整」と「ある程度決まったボールとの相対位置からの一定の打撃動作」の大きく 2 つに分けられると推測される．

そこでまず，人の動作によって直接ロボットを操作する Master-Slave 方式を用いて人間が打撃タスクを行う際の一定の打撃パターンを抽出する．次に，Master-Slave 方式による「位置調整」(バックスイング) 動作から，抽出した「一定の打撃動作」への切替えを含めた打撃実験を行うこのとき，一定動作で打撃が行えるように人が切替えのタイミングをリアルタイムに与えてやり，飛来するボールと切替えの位置やタイミングとの関係をマップとして学習する．最後に，そのマップを用いてバックスイングと一定動作への切替えを予測，実行してロボットによる自律的な打撃動作を実現する．

7.2　Master-Slave を用いたダイナミックマニピュレーション学習実験

Master-Slave を用いたダイナミックマニピュレーション学習システムの構築とその実験は，前節までに述べた人間の計測結果をもとに以下の手順で行うこととした．

- ロボットの制御を「待機動作」と「打撃動作」の 2 パターンに分け，一定パターンの打撃動作を Master-Slave を用いて抽出する．
- 「待機動作」は Master-Slave 方式により追従し，抽出した「打撃動作」への切替えのタイミングを人が何らかの入力を与えて実行する．
- 上記の操作結果をもとに，ボールに対応した動作やタイミングを学習してロボットが自律的に打撃を行う．

7.2.1　Master-Slave による打撃パターン抽出

まずは Master-Slave のみを用いて一定になる打撃パターンを抽出する．実験環境は**図 2.7** に示したように，Master-Slave 方式によりロボットを動作させて打撃を行う．打球機から目標バウンド位置 (300,0)，(500,0) のボールをそれぞれ 1 秒間隔で打ち出し，人 (ロボット) は (-1000,0) を目標に返球した．

図 7.1，**図 7.2** は目標位置の ±100[mm] 以内に打ち返せた時の x 方向の位置および速度の軌道を打撃時刻および打撃位置を 0 として表示したものである．

これらの軌道に対して 7 次式で最小二乗近似を行い，打撃軌道を抜き出した．抜き出したパターンを**図 7.3** に示す．それぞれの式から，600Hz になるように離散的な指令値を作成した．x, y 方向のパターンは指令速度列であり，そのまま 600Hz で指令を送る．$th4(=\theta_4)$ は上下方向の角度パターンで，この角度を追従するように，Master-slave と同様に指令を送る．

7.2.2　打撃パターンへの切替えを含めた打撃によるマップ作成

打撃パターン

Master-Slave を用いて打撃を行った際の実験環境と同じ環境において，前節で抽出した打撃パターン (**図 7.3**) から，前半の一部 (速度コマンドが -1.0[m/s] になるまでの部分) のコマンド列を削除した (理由は後述) パターン (**図 7.4**) を Master-Slave に組み込んで打撃を行った．

打撃パターンへの切替え

打撃パターンへの切替えは，ボタン等で人間が与えるシステムも考えたが，余計な動作を加えると操作する人間に負担を与えることになり，学習そのものに支障が出る可能性が考えられるので，ラケット速度が前方に切り替わった時点で行うこととした．ただし，速度=0 を基準にした場合，微調整のために前方にパドルを移動させることができず，意図しないときに打撃動作に切り替わってしまうという現象が生じた．そこで，人間のパドル速度がある閾値 (-1.0[m/s]) を越えたときに切替えることとし，それにともなって打撃パターンの前半部分の指令速度が -1.0[m/s] 以下の部分をパターンから削除した．この閾値の根拠は，速度指令が -1.0[m/s] のときにロボットのパドルは前方への動作を開始しているからである．

打撃パターンを用いた打撃実験結果

図 7.5 に落下位置をプロットしたものを示した．目標位置から ±100[mm] 以内に落下したものについて，打撃時刻を 0 として，ボールとロボットの x 方向の動作を時間軸に対してまとめて表示したものが**図 7.6** であり，x-y 平面についてまとめたものが**図 7.7** である．打ち返したボールが台上に残った場合に，ボールを取り去る作業を行うために打撃を行わずにボールが通り過ぎることがあったため，打撃率を求める際には，ボールが飛んで来た

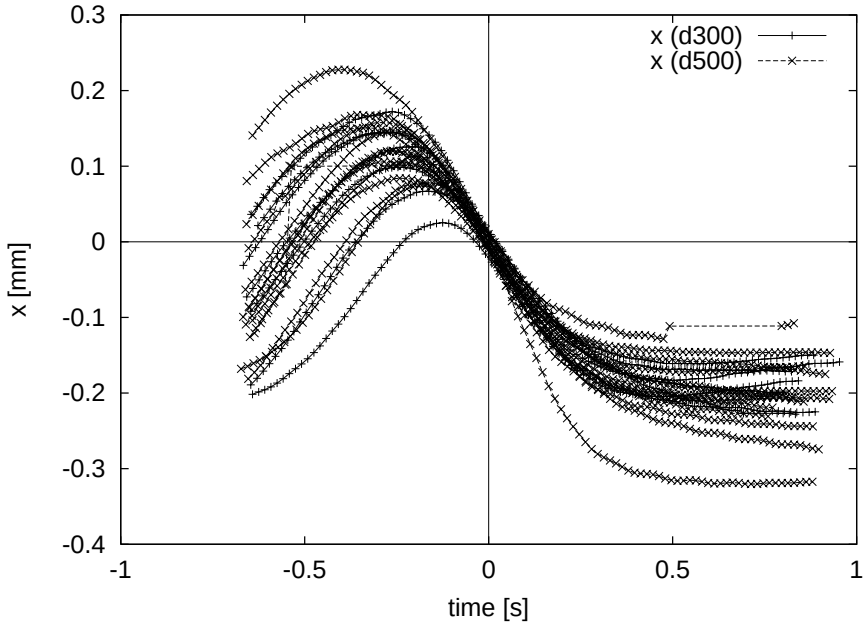


図 7.1 成功時のスイング軌道 (x 方向)

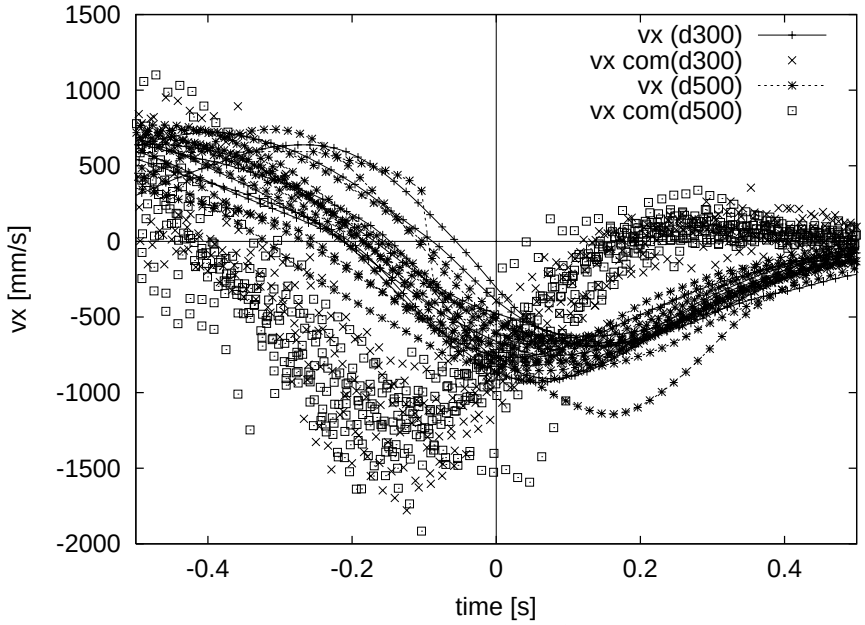


図 7.2 成功時のスイング速度 (x 方向)

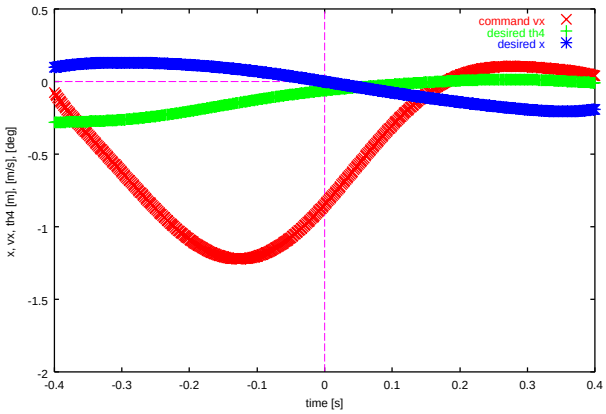


図 7.3　打撃パターン軌道

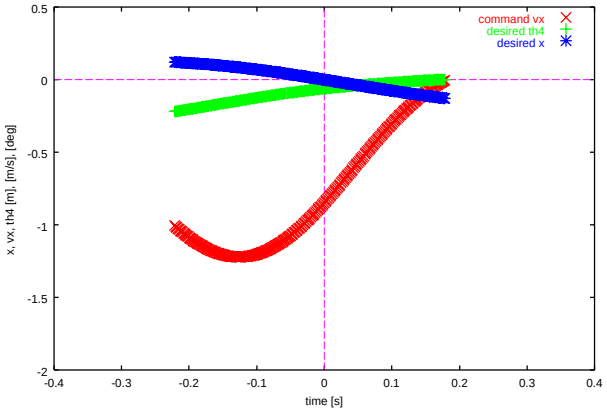


図 7.4　打撃パターン (一部カット)

後にロボットがパターン動作に移行した (打撃動作を行った) 回数に対して，その場合に打ち返した回数を求めた．その結果を表**表 7.2** にまとめる．

また，飛来するボールの相違 (落下位置=300[mm], 500[mm]) に対するバックスイングの時刻および位置の分布をそれぞれ，**図 7.8**，**図 7.9** に示した．

表 7.1　打撃成功率

投入位置	打撃回数	成功回数	打撃率
[mm]	[回]	[回]	[%]
300	325	214	65.58
500	298	198	66.44
Total	623	412	66.13

マップの作成

以上の結果より，返球目標位置の ±100[mm] に返球できた場合のデータをもとに，飛来するボール状態からパターン動作開始点へのマップを作成した．

入力を 5.3 節と同じ，仮想平面上のボール高さ，速度および z 方向加速度とし，出力をバックスイング終了までの時間 $dt = t_b - t_p$ とその時のラケット位置 P_{xb} とするマップを作成した．

$$[p_{bmz}, v_{bmx}, v_{bmy}, v_{bmz}, a_{bmz}] \rightarrow [dt, P_{xb}] \tag{7.1}$$

このマップは，x=-1200[mm] から 100[mm] おきに複数枚用意し，打撃時に正常に計測されたボールデータが十分蓄積された段階で，最も近いマップデータを用いて予測を行う．

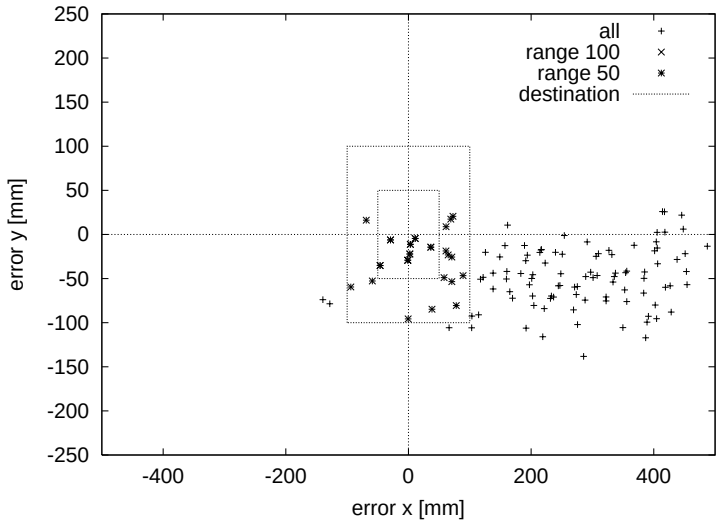


図 7.5 パターン切替え打撃時の落下位置 (目標-1000,0)

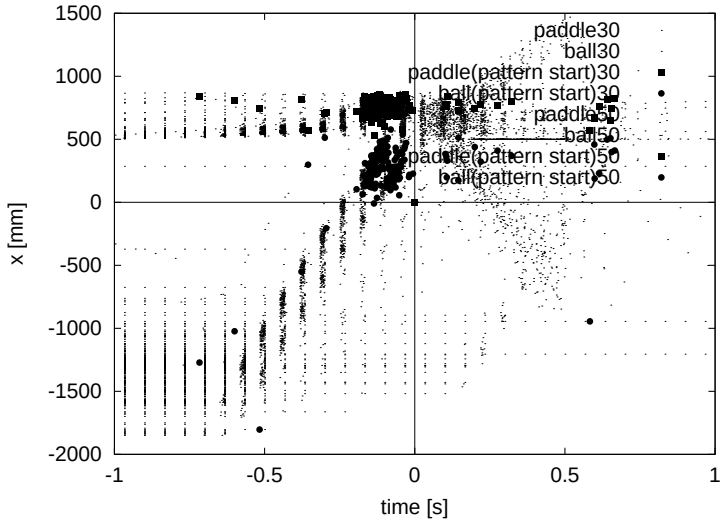


図 7.6 パターン切替え打撃時の軌道 (paddle&ball $t-x$)

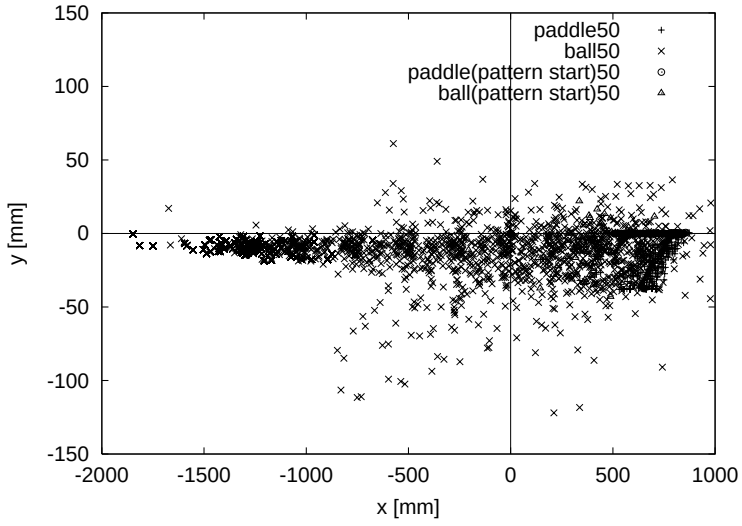


図 7.7 パターン切替え打撃時の軌道 (paddle&ball $x-y$)

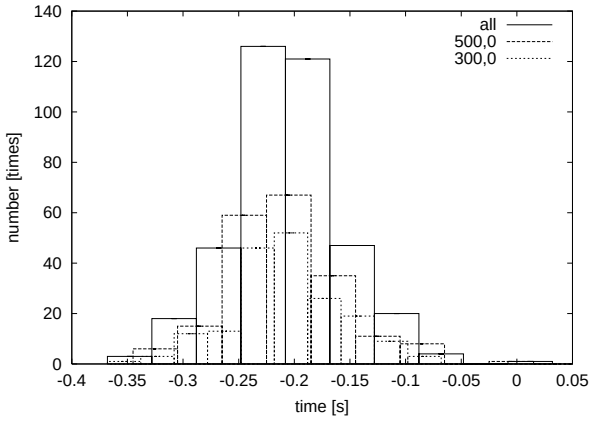


図 7.8 飛来するボールの相違に対するバックスイングの終了時刻の分布

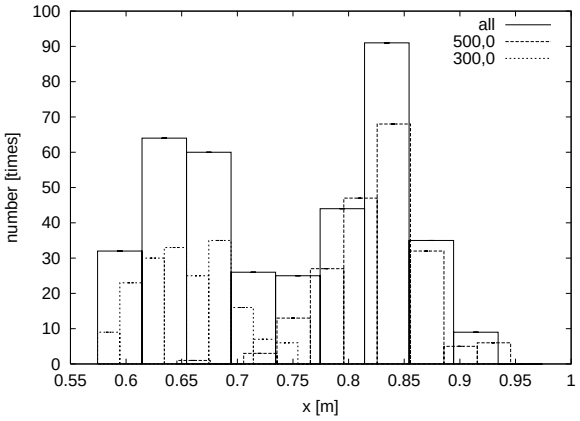


図 7.9 飛来するボールの相違に対するバックスイングの終了位置の分布

7.2.3　作成したパターンとマップを用いた自律制御による打撃

前節で作成されたマップによりバックスイングの終了位置と時刻を予測し，その時刻にその位置まで学習制御の組合せによって移動する．そして，既に得られているパターン動作に切替えて打撃する．このように，人間の制御を完全に排除して打撃実験を行った結果をここに示す.

マップによるバックスイング終了位置の予測

まず，マップを用いた予測方法について説明する．入力のパラメータは 5.3 節におけるマップと同じ ($[p_{bmz}, v_{bmx}, v_{bmy}, v_{bmz}, a_{bmz}]$) であるが，出力値は，仮想平面通過からバックスイング終了までの時間 (dt) とバックスイングの終了位置 (p_{rbx}) である．また，5.3 節におけるマップとのもう一つの相異点は，仮想平面を複数用意した点である．飛んで来たボールが最初のマップ平面を通過し，かつ 7 点以上のボールデータが取得できるまで計測を続ける．その平面のマップにおいて，入力パラメータに近いマップデータが十分存在しない場合は，次の仮想平面で計算し，十分なデータが存在するまでこれを繰り返す．そして予測が出来次第，4 次式軌道を生成してバックスイングを行う.

組合せ学習制御

第 4 章の学習制御手法を用いて，バックスイングを生成する.

指令コマンド列を得るためには，最低 3 つの学習軌道 (動作軌道とコマンド列) を組み合わせる必要があり，前もって通常の学習制御によって求めておく．ここで用いた 3 パターンの軌道を**図 7.10** に示した.

目標バックスイング軌道は，以下のように生成した．現在時刻を $t = 0$ とし，バックスイングの終了時刻を ${}^*t_1$，予測位置を ${}^*x_1$，速度を $v_1 = 0$ とする．この時刻で制御をするには時間が短いので，$t_2 = 2{}^*t_1$ を仮想的な動作終了位置とし，その時刻で加速度 0 となるよ

うに目標軌道を生成する．つまり，時間の 5 次式を求める境界条件として，$t = 0$ のときの $x = 0, vx = 0, ax = 0$，$t = t_1$ のときの $x = {}^*x_1, vx = 0$，そして $t = t_2$ のときの $ax = 0$ を条件として軌道を生成する．

目標軌道の式は，$t = 0$ の初期条件より

$$f_d(t) = c_{d5}t^5 + c_{d4}t^4 + c_{d3}t^3 \tag{7.2}$$

となる．この $\mathbf{C}_d = [c_{d5}, c_{d4}, c_{d3}]^T$ は，残りの条件から次のように求まる．

$$\mathbf{A} = \begin{bmatrix} {}^*t_1^5 & {}^*t_1^4 & {}^*t_1^3 \\ 5{}^*t_1^4 & 4{}^*t_1^3 & 3{}^*t_1^2 \\ 20t_2^3 & 12t_2^2 & 6t_2 \end{bmatrix} \tag{7.3}$$

とすると，

$$\mathbf{A}\mathbf{C}_d = \begin{bmatrix} {}^*x_1 \\ 0 \\ 0 \end{bmatrix} \tag{7.4}$$

より，

$$\mathbf{C}_d = \mathbf{A}^{-1} \begin{bmatrix} {}^*x_1 \\ 0 \\ 0 \end{bmatrix} \tag{7.5}$$

学習済みの軌道の係数をそれぞれ，$\mathbf{C}_a = [c_{a5}, c_{a4}, c_{a3}]^T, \mathbf{C}_b[c_{b5}, c_{b4}, c_{b3}]^T, \mathbf{C}_c = [c_{c5}, c_{c4}, c_{c3}]^T$ として，以下を満たすような k_a, k_b, k_c を求める．

$$\mathbf{C}_d = k_a\mathbf{C}_a + k_c\mathbf{C}_c + k_c\mathbf{C}_c \tag{7.6}$$

この k_a, k_b, k_c を入力列に適用し，目標軌道を実現する新たな入力を生成する．

$$\mathbf{u} = k_a\mathbf{u}_a + k_b\mathbf{u}_b + k_c\mathbf{u}_c \tag{7.7}$$

ただし，$t = 0$ から $t = {}^*t_1$ までの部分のみコマンドを実行する．

スイングパターンとマップを用いた自律打撃実験結果

スイングパターンとマップによるバックスイング終了位置の予測を用いて，自律的な打撃実験を行った結果を示す．**図 7.11** はそのときのボールの動きを示す．下から飛んできたボールが上に抜けている場合は打撃失敗，下に戻っている場合は打撃成功と言える．打撃動作を行った回数が 21 回，打ち返した数は 15 回だったので，打撃成功率は 71.4[%] であった．そのときの予想バックスイング終了位置の分布を加えて，$x - y$ 平面でのボールの軌道を**図 7.12** に示す．しかしながら，バックスイングを生成する際に速度コマンドが指令最大値を越え，動作できない場合が数多く観測された．

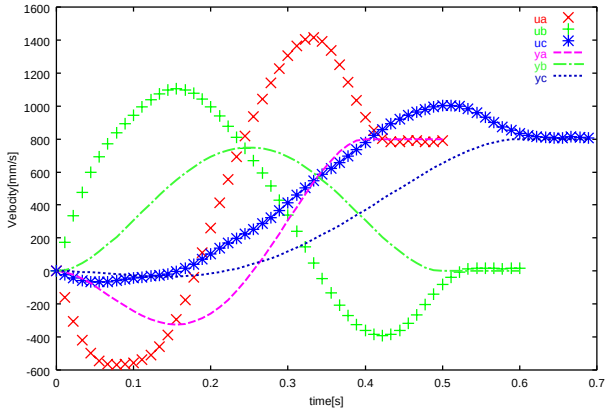


図 7.10　Patterns A, B and C (u*i*: Learned input. y*i*: Desired output.)

7.2.4　考察

人の動作から直接ダイナミックマニピュレーションタスクを実現するシステムの開発の初期段階として，4 自由度のロボットで行った Master-Slave システムを用いたロボットの打撃実験を行った．まず Master-Slave のみを用いた打撃実験により一定パターンの打撃動作を抽出した．次に，抽出された打撃動作と Master-Slave による待機動作 (バックスイング) の組合せによってモデル化が難しかった待機動作のためのマップ作成を行った．さらに，作成したマップを用いて適切なバックスイングの位置とタイミングを予測し，飛んで来るボールに合わせてロボットが自律的に打撃を行う実験を行った．しかしながら，自律動作時のバックスイングの生成時に，指令速度値が最大値を越えたために動作が不可能な場合が多数見受けられた．これは，Master-Slave 制御によるバックスイング動作がボールの見える前に開始しているのに対し，自律動作時のバックスイングはボールが見えてから開始されなければならないという条件の相違によるものと考えられる．そこで，Master-Slave 学習時にボールが見えている場合のみ追従するという拘束を与えることとした．

7.3　拘束を与えたマススレーブ学習

実際の人間の打撃動作をもとにスイングを作成すると，ボールが見えてから動作するロボットでは同様のスイングが不可能だった．そこで，人間が Master-Slave で操作する際の条件に拘束を与えて，ロボットが自律動作する際の条件に合わせることで，効果的な学習結果を得られると考えた．実験の手順の変更点を以下に述べる．

前回の手順,

① Master-Slave による操作のみで打撃を行う

② 上の打撃結果から打撃パターン動作を抽出する．

③ Master-Slave とパターン動作を組み合わせた打撃を行う．

④ 上のときのボールとバックスイング位置のマップを作成する．

⑤ マップでバックスイングを決定し，一定パターンの打撃動作を行う．

このうち，①と③の Master-Slave 動作について，ボールが見えていない場合は人のラケットを追従しないように速度コマンドを送らないこととした．

7.3.1 Master-Slave による打撃パターン抽出実験

実験内容

まずは Master-Slave のみを用いて一定になる打撃パターンを抽出する．実験環境は**図 2.7** に示したように，人はロボットの後方に立ち，人と反対側の集球ネットの裏に設置した打球機から約 1 秒間隔で打ち出されるボールに対し，スイングを行う．ラケットの裏側中央には Fastrak のセンサーを取り付け，Transmitter から発生させられた磁界を検出して人のラケットの 3 次元位置および角度を計測する．この計測値に基づいて，Master-Slave 方式によりロボットを動作させる．ボールの軌道は人の後方に設置された 2 つのカメラ画像を用いて，QuickMAG により計測する．打球機からのボールの目標バウンド位置は (300,0) と (500,0)，(700,0) の 3 種類を用意した．また，人 (ロボット) の返球目標位置は (1000,0) とした．(**図 2.7**)

実験結果

図 7.1 は目標位置の ±100[mm] 以内に打ち返せた時の x 方向および y 方向のラケットの軌道を，打撃時刻および打撃位置を 0 として表示したものである．

これらの軌道に対して 7 次式で最小二乗近似を行い，打撃軌道を抜き出した．抜き出したパターンを**図 7.3** に示す．それぞれの式から，600Hz になるように離散的な指令値を作成した．x, y 方向のパターンは指令速度列であり，そのまま 600Hz で指令を送る．$th4(= \theta_4)$ は上下方向の角度パターンで，この角度を追従するように，Master-slave と同様に指令を送る．

7.3.2 打撃パターンへの切替えを含めた打撃によるマップ作成 (拘束有り)

打撃パターン (拘束有り)

拘束なしの場合と同様に，Master-Slave のみの動作結果から抽出した打撃パターン (**図 7.15**) の，前半の一部を削除したパターン (**図 7.16**) を Master-Slave に組み込んで打撃を行った．

打撃パターンを用いた打撃実験結果 (拘束有り)

拘束なしの場合と同様にパターンへの切替えを行い打撃実験を行った．**図 7.17** に落下位置をプロットしたものを示した．目標位置から ±100[mm] 以内に落下したものについて，

打撃時刻を 0 として，ボールとロボットの x 方向の動作を時間軸に対してまとめて表示したものが**図 7.6** であり，$x-y$ 平面についてまとめたものが**図 7.7** である．打撃率を求める際には，ボールが飛んで来た後にロボットがパターン動作に移行した (打撃動作を行った) 回数に対して，その場合に打ち返した回数を求めた．その結果を表**表 7.2** にまとめる．

また，飛来するボールの相違 (落下位置=300[mm], 500[mm]) に対するバックスイングの時刻および位置の分布をそれぞれ，**図 7.20**，**図 7.21** に示した．

以上の結果より，返球目標位置の ±100[mm] に返球できた場合のデータをもとに，7.2.2 節と同様のマップを作成した．

表 7.2　打撃成功率

投入位置 [mm]	打撃回数 [回]	成功回数 [回]	打撃率 [%]
300	243	156	64.20
500	221	171	77.38
700	206	167	81.07
Total	670	494	73.73

7.3.3　拘束を加えて作成したパターンとマップを用いた打撃実験

拘束を加えて抽出したスイングパターンと前節で作成したマップによるバックスイング終了位置の予測を用いて，打撃実験を行った結果を示す．**図 7.22** はボールの軌道の一部を示す．予測を行い，実際に動作した回数は 708 回，打ち返した数は 618 回だったので，打撃成功率は 87.28[%] であった．

そのときの予想バックスイング終了位置の分布を加えて，x-y 平面でのボールの軌道を**図 7.23** に示す．

7.4　仮想マップによる打撃実験

前回までの実験で作成したパターンと新たに作成したマップを用いて自律制御による打撃を行った．今回は，3 種類 (目標落下位置 300mm,500mm，700mm) のボールを供給し，各ボールに対する出力を一定値とした仮想的なマップを作成した．

そのマップを用いて，新たに供給したボールに対するバックスイングの終了位置と時刻を予測し，その時刻にその位置まで学習制御の組合せによって移動する．そして，既に得られているパターン動作に切替えて打撃する．このようにして，人間の制御を完全に排除した打撃実験を行った．その結果をここに示す．

7.4.1 仮想マップ

仮想マップの作成は以下の手順で行った.

1. 3 種類 (目標落下位置 300mm,500mm，700mm) のボールを供給して軌道を計測する.

2. 計測した軌道をもとにマップの入力値を集め，それぞれの目標落下位置に対するボールの入力ベクトル (z, vx, vz, az) を求める.

3-1. 2 の入力ベクトルに対する出力はそれぞれの打球機の目標落下位置から $200mm$ 後ろに設定した.

3-2. 時間の出力はボールの打撃想定位置 ($z = 200$mm) 通過時刻とした.

図 7.24〜**図 7.30** にそれぞれのボールに対するマップの入力とマップのイメージを示す.

7.5 クロスバリデーションエラーチェックによるマップの外れ点除去

7.5.1 外れ点を除去したマップ

本来のデータからノイズやバラツキを排除するために CrossValidationError を用いて，平坦にしたマップデータ (**図 7.32**, **図 7.33**) を用いて自律打撃実験を行った．具体的に，LWR 版の leave-one-out cross validation error は,

$$r_i^{CV} = \frac{w_i(y_i - \boldsymbol{x_i}^T\boldsymbol{\beta})}{1 - \boldsymbol{z_i}^T(\boldsymbol{Z}^T\boldsymbol{Z})^{-1}\boldsymbol{z_i}} \tag{7.8}$$

と表され, その 2 乗平均は

$$MSE^{CV}(q) = \frac{1}{n_{LWR}}\sum_{i=1}^{n}(\frac{w_i(y_i - \boldsymbol{x_i}^T\boldsymbol{\beta})}{1 - z_i^T(Z^TZ)^{-1}z_i})^2 \tag{7.9}$$

$$n_{LWR} = \sum_{i=1}^{n} w_i^2 \tag{7.10}$$

となる.

この平均二乗誤差 ($MSE^{CV}(q)$) を各 query 点について調べ,「自分自身を除いたそれ以外のデータによって推定した値と自分自身との誤差が，平均二乗誤差の 2 倍を越えるようなら，そのデータを排除する」という方法で，外れ点の除去を行った.

7.5.2　打撃実験結果

加工されたマップを用いて自律打撃実験を行ったときの予測結果は，**図 7.35**，**図 7.36** のようになった．そのときの入力分布は**図 7.37** である．全ての打撃におけるボールの軌道を**図 7.38**，**図 7.39** に示す．また，1 回の打撃結果を**図 7.40** に示す．予測ができて実際に動作した回数は 81 回，打ち返した回数 78 回．ヒット率は 96.3[%] であった．

ただし，**図 7.41** に示すように，落下位置は全て短く，目標の-1000[mm] に届かなかった．ボールの制御は不十分だが，人間の動的スキルを直接ロボットに教示する Master-Slave システムの開発を行い，限定された条件ではあるが実現した．

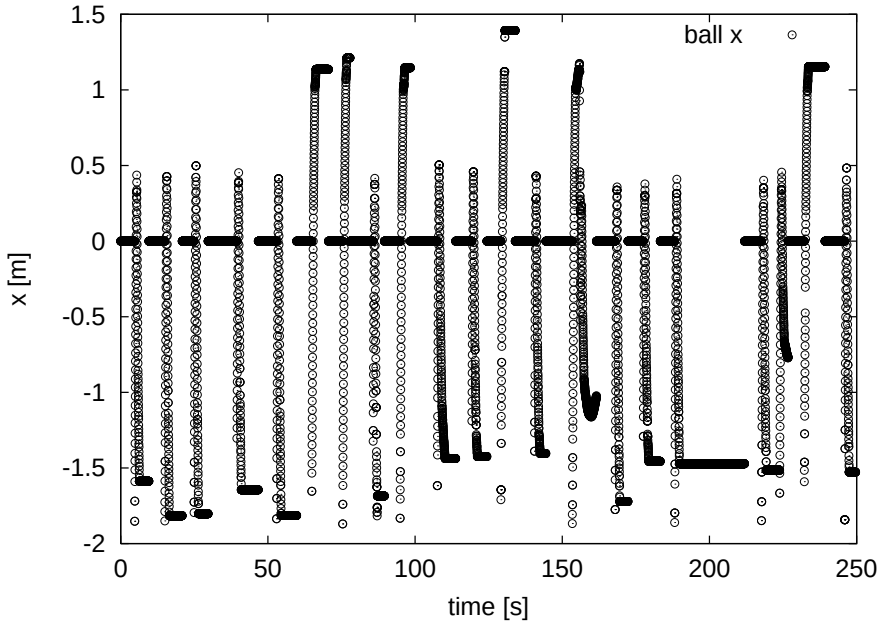


図 7.11　自律打撃実験時のボール軌道 (x)

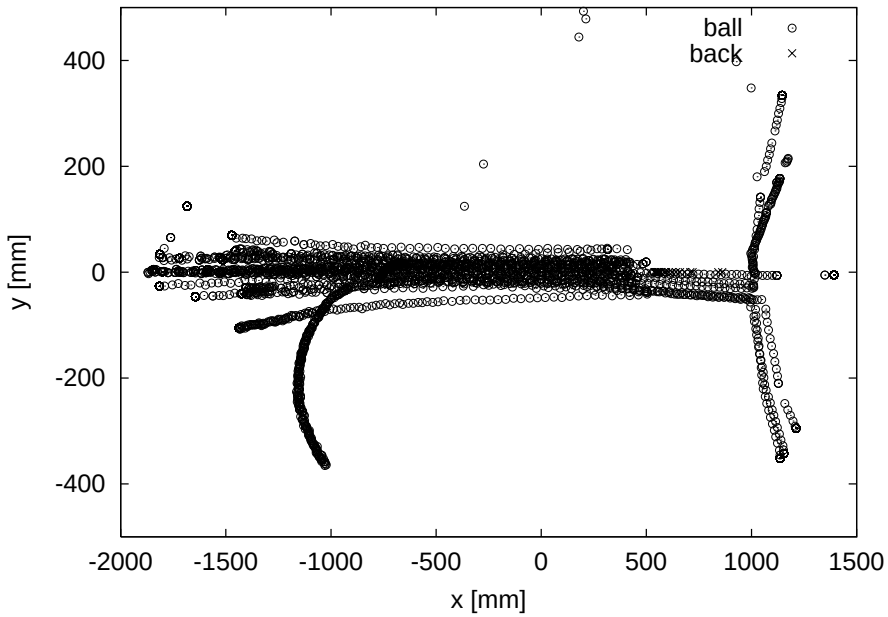


図 7.12　ボール軌道とバックスイングの予測位置 $(x$-$y)$

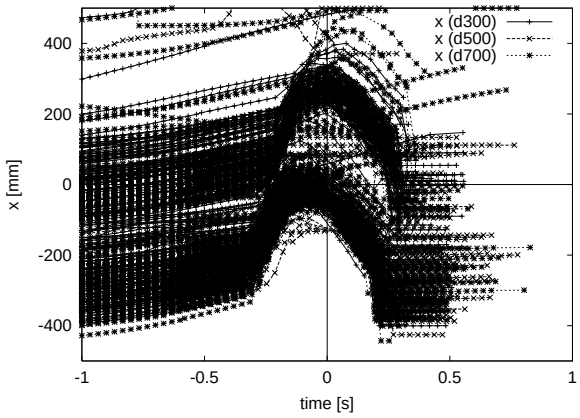


図 7.13 成功時のスイング軌道 (x 方向)(拘束有)

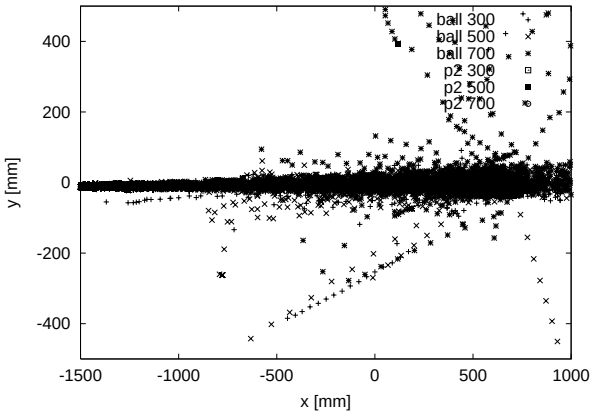


図 7.14 成功時のスイング軌道 (x-y 平面)(拘束有)

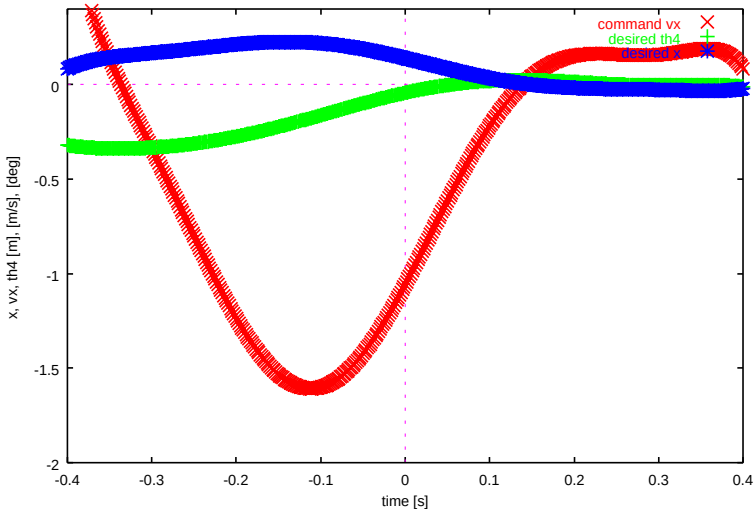


図 7.15　打撃パターン軌道 (拘束有り)

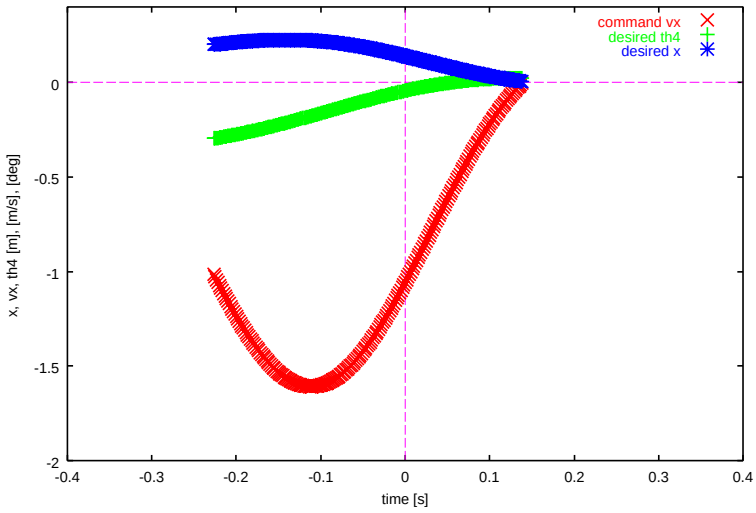


図 7.16　打撃パターン (拘束有り，一部カット)

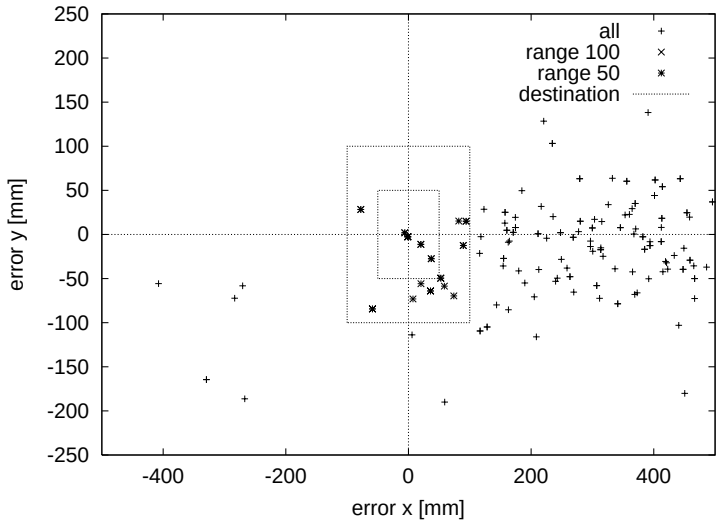


図 7.17 パターン切替え打撃時の落下位置 (目標-1000,0)(拘束有)

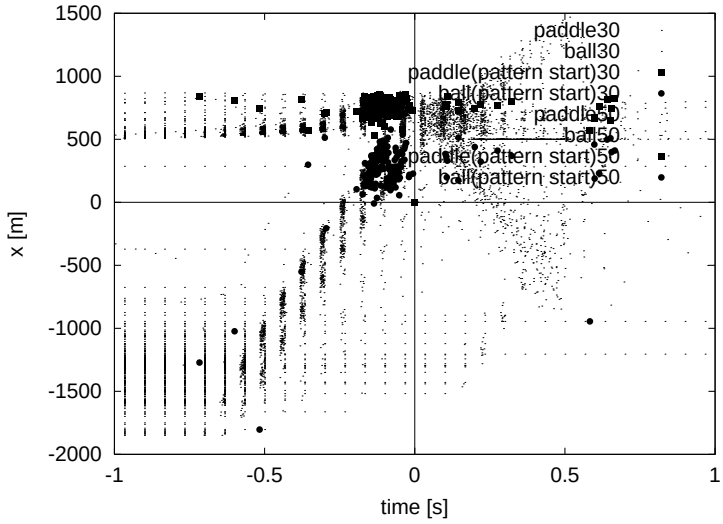


図 7.18 パターン切替え打撃時の軌道 (paddle&ball $t-x$)(拘束有)

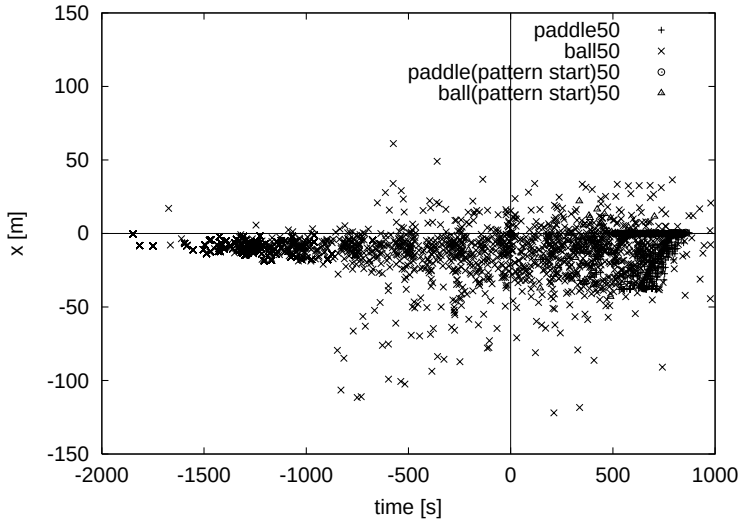


図 7.19 パターン切替え打撃時の軌道 (paddle&ball $x-y$)(拘束有)

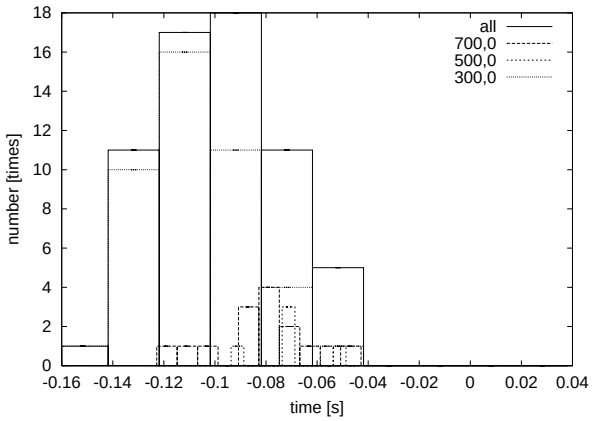


図 7.20 飛来するボールの相違に対するバックスイングの終了時刻の分布 (拘束有)

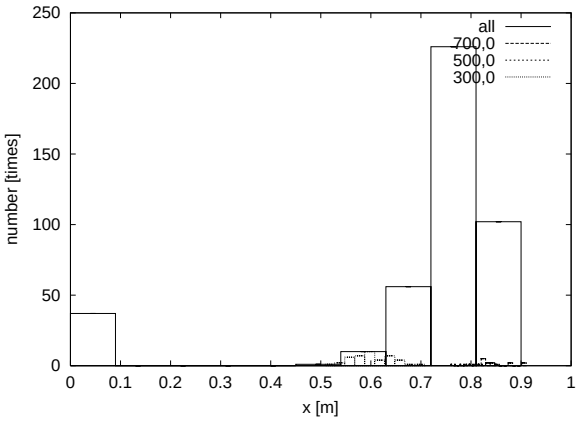


図 7.21 飛来するボールの相違に対するバックスイングの終了位置の分布 (拘束有)

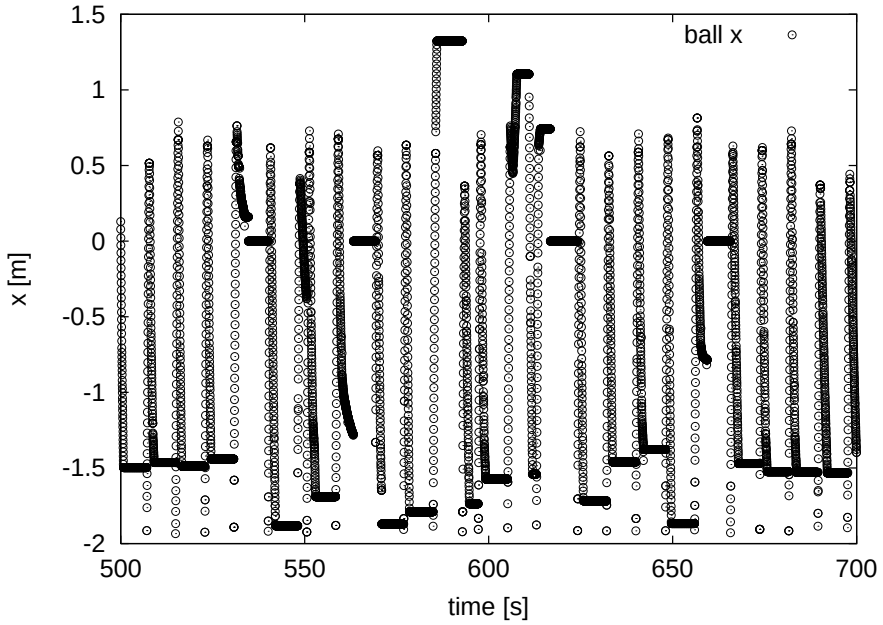


図 7.22　自律打撃実験時のボール軌道 (x)(拘束有)

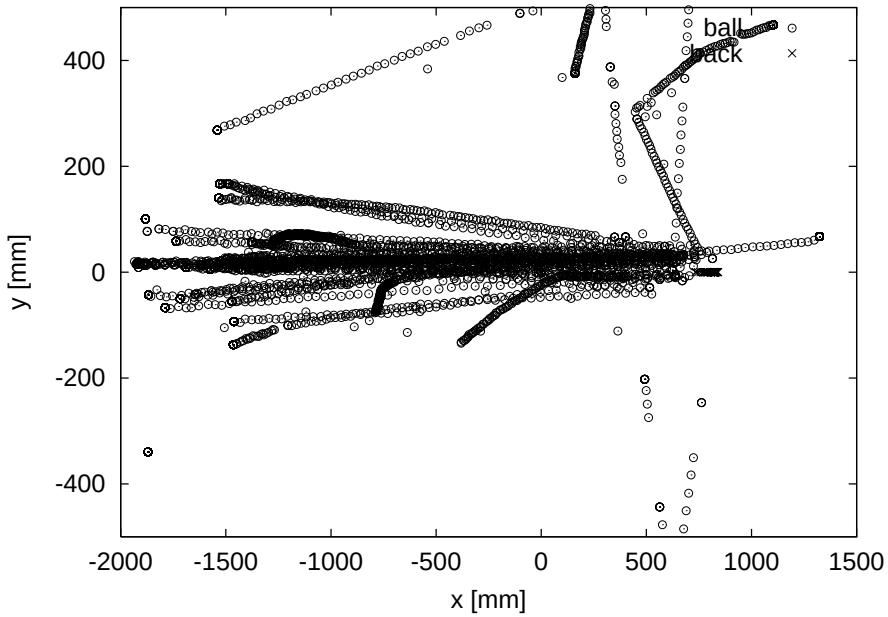


図 7.23　ボール軌道とバックスイングの予測位置 $(x$-$y)$(拘束有)

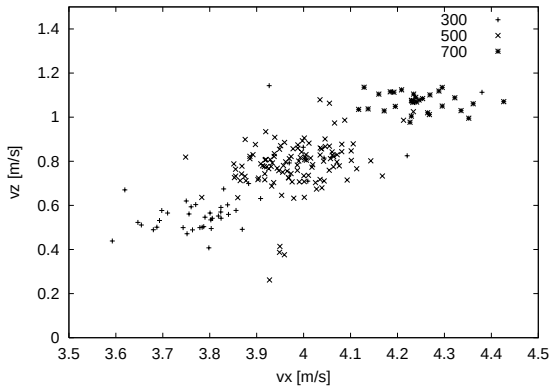


図 7.24　マップ入力 (vx-vz)

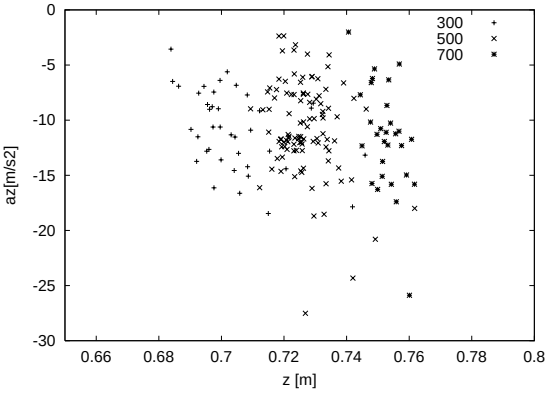


図 7.25　マップ入力 (z-az)

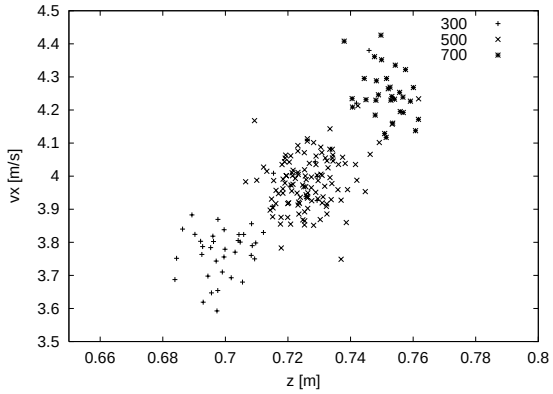


図 7.26　マップ入力 (z-vx)

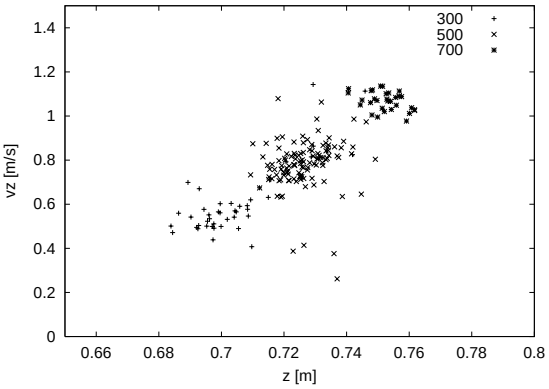


図 7.27　マップ入力 (z-vz)

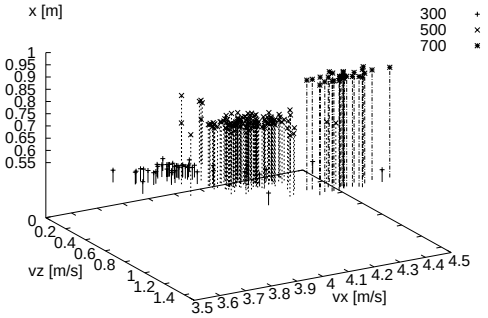


図 7.28　x マップ (vx-vz)

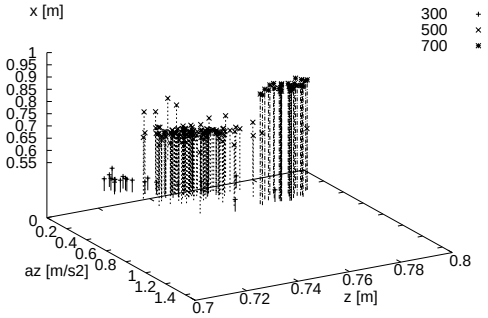


図 7.29　x マップ (vx-vz)

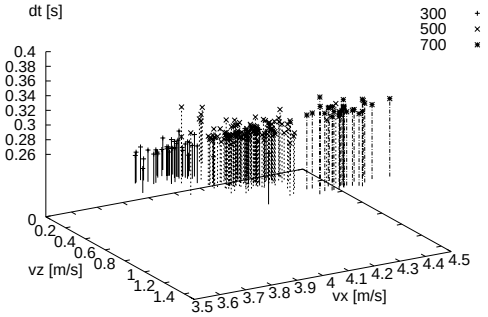


図 7.30　t マップ (vx-vz)

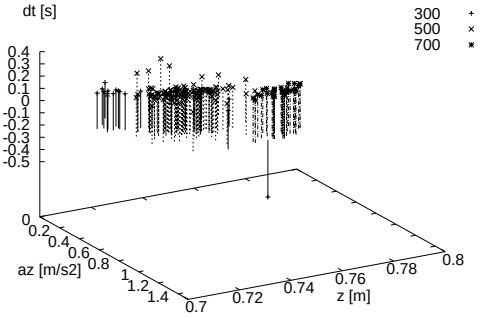


図 7.31　t マップ (vx-vz)

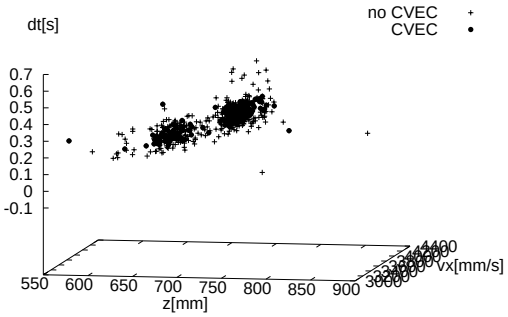


図 7.32 t マップ (Cross Validation Error Check)

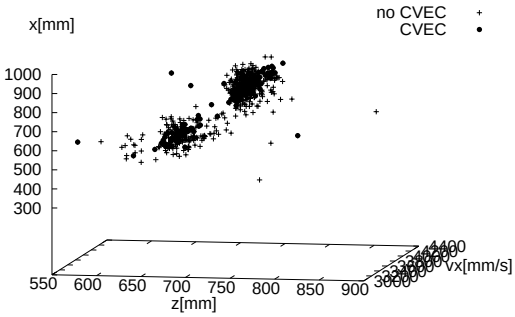


図 7.33 x マップ (Cross Validation Error Check)

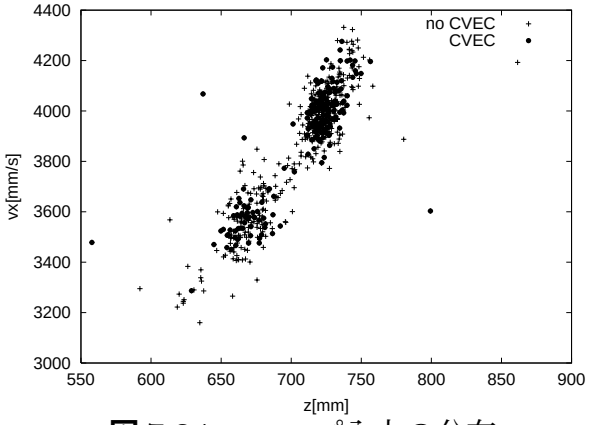


図 7.34 マップ入力の分布

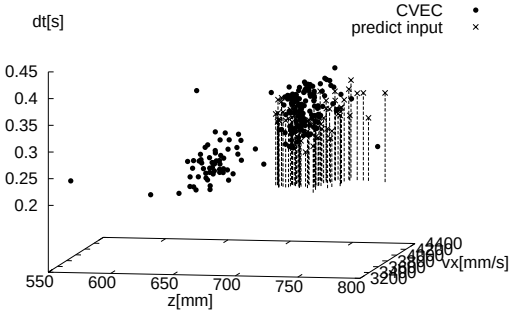


図 7.35 dt マップによる予測結果 (CVEC マップ)

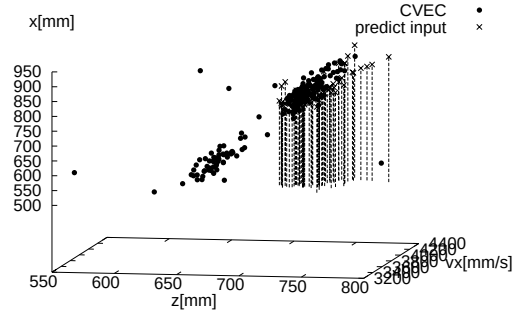


図 7.36 x マップによる予測結果 (CVEC マップ)

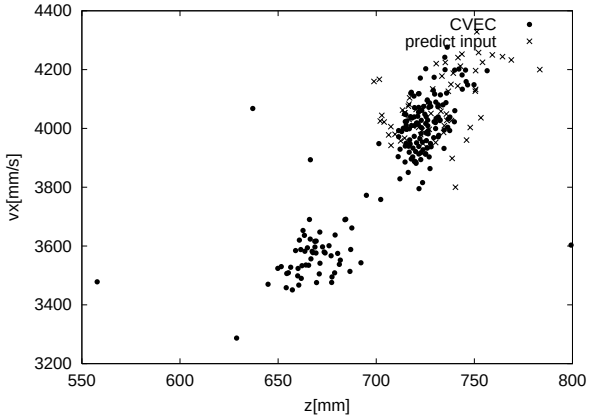


図 7.37 予測時のマップ入力の分布 (CVEC マップ)

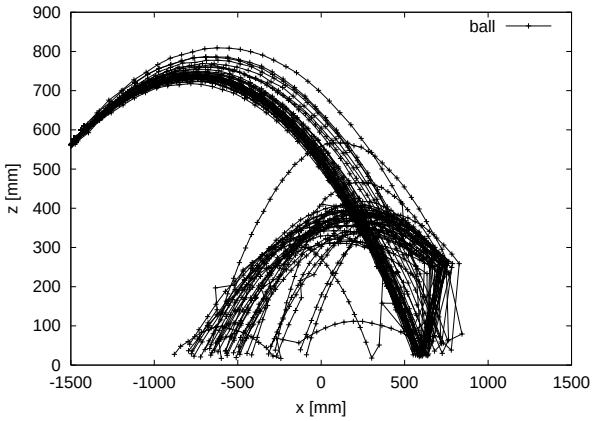


図 7.38　ボールの軌道 x-z(CVEC マップ)

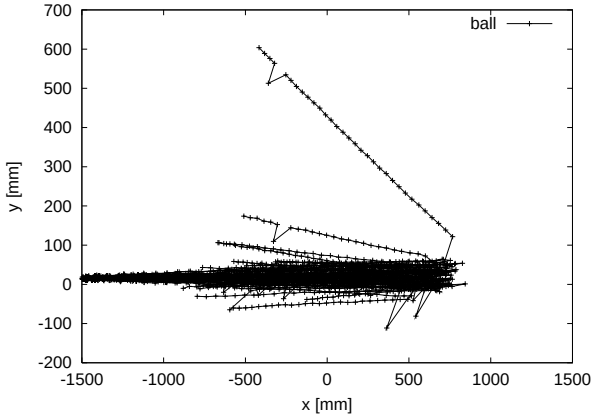


図 7.39　ボールの軌道 x-y(CVEC マップ)

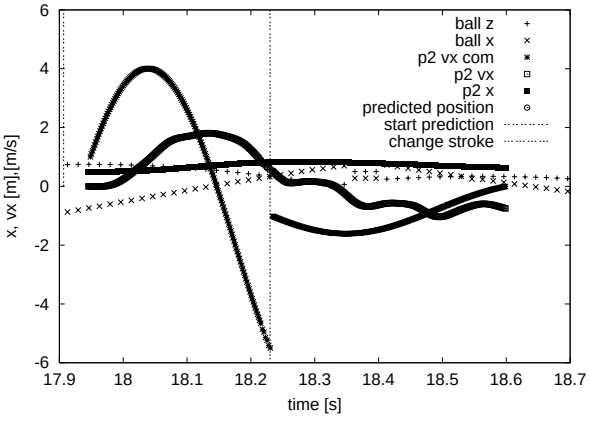


図 7.40　自律打撃 (CVEC マップ)

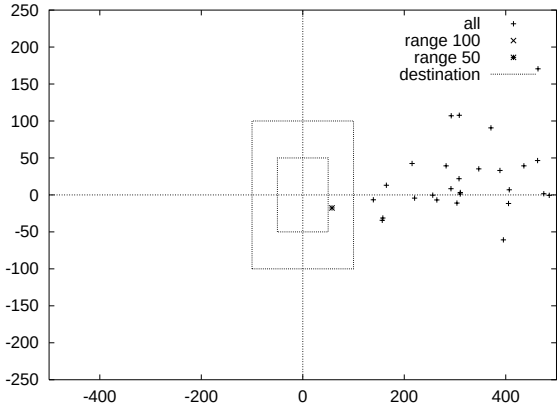


図 7.41 自律打撃落下位置 (CVEC マップ)

第 8章　結論

ロボットによる動的ターゲットに対する対応能力を開発するため，卓球タスクを取り上げて研究を行った．まず，飛来するボールを適切に打ち返すラケットの打撃条件を仮想ターゲットと呼び，これを実現することで卓球における打撃を行うシステムを，ミラー理論に類似の視覚フィードバック制御と入出力マップを用いた予測によって実現した．本論文では，卓球タスクにおけるストローク動作の主要部分は仮想ターゲットの予測に基づいて実行されるものと考え，その予測のための２つの入出力マップを提案した．その１つである打撃位置決定用マップによってボールの打ち返し位置が決定され，もう１つのミラーゲイン決定用マップによって打撃時のラケットスピードに対応したミラーゲインが決定される．決定されたこれらの仮想ターゲットをミラー則に類似の視覚フィードバック則を用いてラケットの動作に反映させ，ボールをラケットでとらえ目標の着地点付近に打ち返すことができることを示した．ただし，本論文で実現した卓球タスクは必要最小限のラケット自由度と制御パラメータを用いて行ったものであり，打ち返し可能なボールの速度やコースが限定される．また，サーボ系の遅れやシステムの弾性要素により指令値と実際の軌道に遅れが生じていたために，不安定な軌道を生成していた．

これらの問題点を解決するため，新しい制御手法を提案した．我々の使用するロボットは，トルクが小さく，時間遅れが大きいため，打撃動作中での大きな軌道更新は困難であり，頻繁な軌道更新を行う制御は適さない．また，0.4(sec) 程度の短時間での高速動作を想定しているため，フィードフォワード方式の制御が適していると考えた．CP(continuous path) 制御を実現するフィードフォワード制御入力を生成する方法として学習制御 (ILC) があり，制御対象のダイナミクスを厳密に推定することなく，目標軌道を実現するフィードフォワード入力を獲得できる利点があるが，通常の学習制御では学習に数回から数十回の繰り返し動作を要するため，単一軌道の実現に用いられるのが一般的であり，ダイナミックマニピュレーションのように，環境変化にともなって毎回必要な動作軌道が変化する場合には適さない．そこで，目標軌道が多項式で表されていることとロボットの線形性を利用して，未学習の軌道を正確に実現する入力パターンを繰返し学習を行わずに求める Direct ILC の手法を提案し，実機によってその有効性を確かめた．

次に，仮想ターゲットの考え方とミラー理論を応用して実現した卓球タスクは必要最小限のラケット自由度と制御パラメータを用いて行ったものであり，打ち返し可能なボールの速度やコースが限定されていたため，より柔軟にボールを返球できるシステムを開発した．まず，卓球タスクにおける 3 つの物理現象を入出力マップによって表現し，マップ 1 による打撃時のボール状態を予測し，マップ 2，マップ 3 の逆マップを利用して目標の飛行距離と飛行時間を実現するためのラケット速度およびラケット角度を仮想ターゲットとした．マップの実装は LWR によって行った．LWR マップを用いて作成した動作計画を Direct ILC によって正確に実現することで，ボールの飛距離と飛行時間を制御して返球するボー

ル操作タスクを実現した.

また，ロボットが人間にとって打ちやすいボールを，人間がロボットに打ちやすいボールを打つことで，ラリータスクを行った．ロボットは飛来するボールの状態と目標から打撃時のラケット状態を決定し，そのラケット状態を正確に実現する打撃動作を生成することで適切なボールの返球を可能とした．また，1 行程平均約 1.5[s] という短時間の打ち返し動作を繰り返し行うことにより，対人のラリーを継続させることに成功した．人間の打ち出す，毎回異なるボールに対して，ロボットは相手コートの目標付近にボールを返球することに成功し，ラリータスクを実現した.

人間のダイナミックマニピュレーション動作をロボットの制御に適用するために，人間の運動や環境変化を表す対象物体 (卓球タスクにおけるボール) の運動を制御対象となるロボットの運動にマッピングすることを目的として，手先や身体の位置および速度を計測した．その中で，ボールの運動と関連性のある動作とそうでない動作を，分類し，基本動作と環境変化への対応動作を別々に扱うことによって，ロボットの対応能力の実現手法の開発を目指した.

人間の動作をロボットの制御に適用するために，人間の動作の計測を行った結果，以下の特徴が得られた.

① バックスイングがボールに合わせて大きくばらつくのに対し，フォワードスイングはばらつきが少ない.

② 飛来するボールの速度・打撃目標位置・打撃目標速度の 3 つから，バックスイング終了の位置とタイミングを決定.

③ バックスイング終了から打撃にかけての運動を同じタイミングで行っている

④ 打撃の瞬間については短いボールに対して返球する場合も含めて，全ての場合で加速度が 0 付近になっている.

これらをもとに，以下のような戦略で人がタスクを実現する際のタイミングや動作パターンおよびボールの挙動を，ボールの速度や位置などの情報とやタイミングや待機位置との関係を表すマップとして直接ロボットが学習しながらダイナミックマニピュレーションタスクの実現を目指した.

- ロボットの制御を「待機動作」と「打撃動作」の 2 パターンに分け，一定パターンの打撃動作を Master-Slave を用いて抽出する.
- 「待機動作」は Master-Slave 方式により追従し，抽出した「打撃動作」への切替えのタイミングを人が何らかの入力を与えて実行する.
- 上記の操作結果をもとに，ボールに対応した動作やタイミングを学習してロボットが自律的に打撃を行う.

ボールの制御は十分とは言い難いが，人間の動的スキルを直接ロボットに教示し，人間が学習によって身につけたスキルをロボットが利用して，飛んで来るボールに合わせた打撃タスクを実現した．複雑な動作計画を与えてやることなく直接的なマスタースレーブ学習によってダイナミックマニピュレーションスキルをロボットに実装する学習教示手法の実用の可能性を示した．人間のスキルを実現しやすい機構のロボットを用いて，より複雑なスキルの学習を実現することが今後の課題となる．

付録 A　Locally Weighted Regression(LWR)

ここでは, 一般的な線形回帰と LWR による予測方法を紹介し, 回帰分析の評価方法である Leave-one-out cross validation(LOOCV) を用いた LWR のパラメータ設定方法, 及びデータベース内の, はずれ点除去方法について述べる [23, 29, 11, 30, 31].

A.1　一般的な線形重回帰

p 個の説明変数 $x_1, ..., x_j, ..., x_p$,1 個の目的変数 y を持つ n 個のサンプルデータを関数

$$y = \beta_0 + \beta_1 x_1 + ... + \beta_p x_p \tag{A.1.1}$$

で近似する時, 最小 2 乗法では,

$$C = \sum_{i=1}^{n} (\boldsymbol{x}_i^T \boldsymbol{\beta} - y_i)^2 \tag{A.1.2}$$

なる評価関数 C を最小にする係数ベクトル $\hat{\boldsymbol{\beta}}$ を $\partial C/\partial \beta = 0$ により決定する. ここで x_i,y_i は i 番目のサンプルデータで, $\boldsymbol{x_i}$ =$[1., x_{i1}, ..., x_{ij}, ..., x_{ip}]^T$ である.

行列で表すと β は

$$\hat{\boldsymbol{\beta}} = (\boldsymbol{X}^T \boldsymbol{X})^{-1} \boldsymbol{X}^T \boldsymbol{y} \tag{A.1.3}$$

により求まる. ここで $\boldsymbol{X}$ は i 列が $\boldsymbol{x_i}^T$ の $n \times (p+1)$ の行列であり, $\boldsymbol{y}$ は i 番目の要素が y_i のベクトルである.

query 点の予測値は $\hat{\boldsymbol{\beta}}$ を用いて,

$$\hat{y}(\boldsymbol{q}) = \boldsymbol{q}^T \hat{\beta} \tag{A.1.4}$$

により決定される.

A.2　LWR

LWR では,query 点 $\boldsymbol{q}$ と各データ点との距離に応じた重み付けを行い回帰を行う. $d(\boldsymbol{x}_i, \boldsymbol{q})$ を q 点と $\boldsymbol{x}_i$ 点の距離とし, 重み関数を $K(d(\boldsymbol{x}_i, \boldsymbol{q}))$ とすると, 評価関数

$$C(q) = \sum_{i=1}^{n} [(\boldsymbol{x}_i^T \beta - y_i)^2 K(d(\boldsymbol{x}_i, \boldsymbol{q}))] \tag{A.2.5}$$

を最小にする $\hat{\boldsymbol{\beta}}$ を決定し,

$$\hat{y}(\boldsymbol{q}) = \boldsymbol{q}^T \hat{\boldsymbol{\beta}} \tag{A.2.6}$$

を予測値とする.

重み関数の選び方には様々なものがあるが, ここで用いるモデルは, 入力点 q と i 番目のデータ x_i との距離関数 d_i を式 (A.2.7), x_i の重み関数 $K(d_i)$ を式 (A.2.8) とする. ここで m_j は j 軸方向の距離に関する重みを表し, h は距離と重みの関係を決定する.

$$d_i = \sqrt{\sum_{j=1}^{p} m_j^2 (x_{ij} - q_j)^2} \tag{A.2.7}$$

$$K(d_i) = w_i^2 = exp(-\frac{d_i^2}{h^2}) \tag{A.2.8}$$

w_i を用いて式 (A.2.5) を書き直すと,

$$C(q) = \sum_{i=1}^{n} [(\boldsymbol{x}_i^T \beta - y_i)^2 w_i^2] \tag{A.2.9}$$

のように表され, 対角成分が w_i の $n \times n$ 行列 $\boldsymbol{W}$ を用いて $\boldsymbol{Z} = \boldsymbol{W}\boldsymbol{X}$,$\boldsymbol{v} = \boldsymbol{W}\boldsymbol{y}$ のように表すと, 式 (A.1.3) と同様, 式 (A.2.10) のように行列演算の形で β が表せる.

$$\hat{\beta} = (\boldsymbol{Z}^T \boldsymbol{Z})^{-1} \boldsymbol{Z}^T \boldsymbol{v} \tag{A.2.10}$$

これより q 点での予測値は,

$$\hat{y}(\boldsymbol{q}) = \boldsymbol{q}^T (\boldsymbol{Z}^T \boldsymbol{Z})^{-1} \boldsymbol{Z}^T \boldsymbol{v} \tag{A.2.11}$$

となる.

A.2.1 LOOCV

LWR 版の leave-one-out cross validation error は,

$$r_i^{CV} = \frac{w_i (y_i - \boldsymbol{x_i}^T \boldsymbol{\beta})}{1 - \boldsymbol{z_i}^T (\boldsymbol{Z}^T \boldsymbol{Z})^{-1} \boldsymbol{z_i}} \tag{A.2.12}$$

と表され, その 2 乗平均は

$$MSE^{CV}(q) = \frac{1}{n_{LWR}} \sum_{i=1}^{n} (\frac{w_i (y_i - \boldsymbol{x_i}^T \boldsymbol{\beta})}{1 - z_i^T (Z^T Z)^{-1} z_i})^2 \tag{A.2.13}$$

$$n_{LWR} = \sum_{i=1}^{n} w_i^2 \tag{A.2.14}$$

で表される [11].

A.2.2 距離関数の決定方法

データ全体を一つのモデルで表す方法と異なり, LWR では入力データと距離が近いデータに, より大きな重みづけを行ない出力を決定するため, 各入力軸方向の距離を均等に扱うのではなく,「距離=性質の違い」となるように距離関数を設定する必要がある. つまり, 式 (A.2.7) の距離関数の重み m_j の設定が重要になる. そこで次のような評価関数 $C(M)$ を用意し, この $C(M)$ を最小にするような M をデータベース内のデータを用いてあらかじめ決定しておく. M は対角成分が m_j の (p+1)×(p+1) の行列.

$$C(M) = \sum_{i=1}^{n} MSE^{CV}(x_i; M) \tag{A.2.15}$$

A.2.3 重み関数の決定方法

式 (A.2.8) の h は距離と重みの関係を決定する. h が小さいという事は入力点に距離の近いデータだけ参照する事を意味し, 逆に h が大きいという事は遠くのデータもある程度参照する事を意味する. この h は入力近傍のデータ密度に応じて決定する事が望ましいので, 式 (A.2.13) を最小にする h を入力 q が与えられる毎に決定し, 式 (A.2.11) により予測出力を決定する. ただし,h を小さくしすぎるとオーバーフィッティングの恐れがある事と, $|W|$ が小さくなり数値計算に問題がでるため最低値を設けている.

A.2.4 はずれ点除去方法

データベースには, 計測誤差の大きなデータや性質が大きく異なるデータも含まれる. こういったデータは予測結果に悪影響を及ぼすため, 予測に先駆けてデータベースから除去したり, 予測時にその性質の違いを判断し予測に用いないようにする必要がある. そこで, 十分な数のデータを取得後に, 予測に先駆けた前処理として次のような基準でデータを除去する事とした.

式 (A.2.13) で表される各点中心の $MSE^{CV}(x_i)$ を求め, その全データの分散 σ に対して

$$MSE^{CV}(x_i) > k\sigma \tag{A.2.16}$$

となるデータ x_i をはずれ点として除去する.

A.3 簡単な関数を用いた検証

2 入力 1 出力の関数

$$y = x_1 + x_1x_2 + x_2^2 \tag{A.3.17}$$

を用いて LWR の予測精度検証を行った.

サンプルデータ作成

式 (A.3.17) から 100 点のサンプルデータを作成した. 作成手順は以下の通りである.

① 標準偏差 1.0 の標準分布からランダムに x_{1i},x_{2i},x_{3i} を決定する.

② $y_i = (x_{1i} + \epsilon_{1i}) + 2(x_{2i} + \epsilon_{1i})^2 + \epsilon_{2i}$ より y_i を決定する.

③ i=100 まで繰り返す.

ここで ϵ_{1i} は標準偏差 0.01 の分布から,ϵ_{2i} は標準偏差 0.05 の分布からランダムに発生させた. また,x_3 は y_i に無関係な変数として用意した.

図 A.3.1∼A.3.5 にサンプルデータの分布を示す.

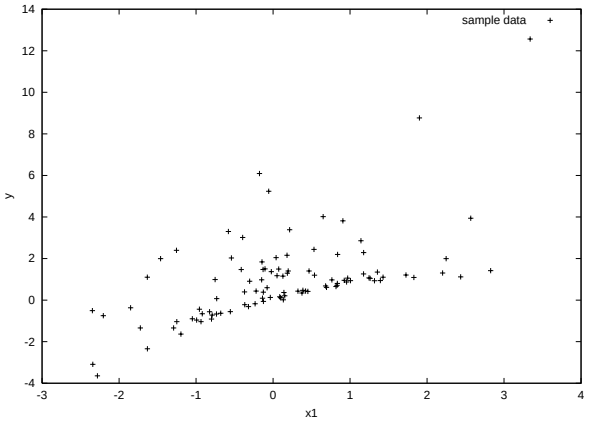


図 A.3.1　データの分布 (x_1-y)

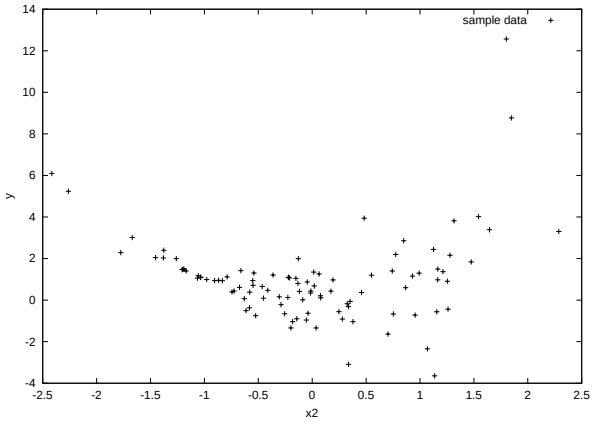


図 A.3.2　データの分布 (x_2-y)

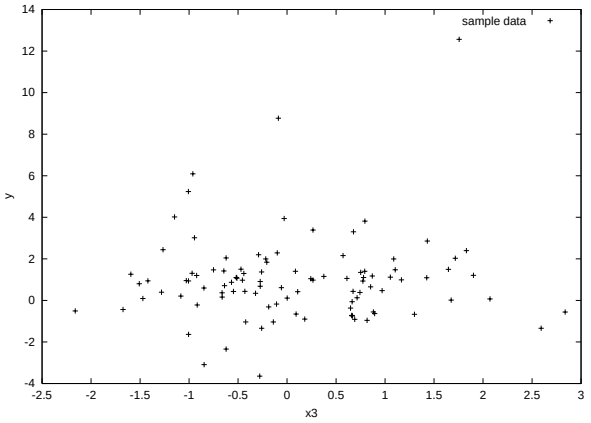


図 A.3.3　データの分布 (x_3-y)

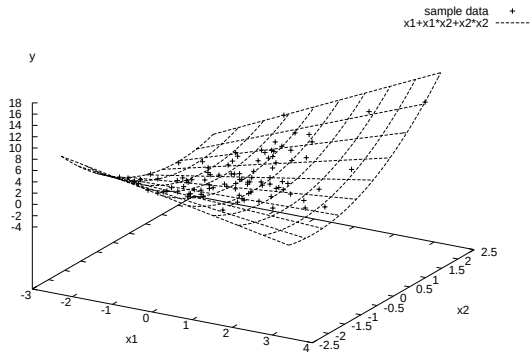


図 A.3.4　データの分布 (x_1-x_2-y)

距離関数の決定

式 (A.2.15) で表される評価関数を最小にする距離関数の重み行列 M を修正パウエル法 [32] を用いて決定した結果, $m_1/|M| = 0.189, m_2/|M| = 0.982, m_3/|M| = 0.029, h = 0.202$ であった.

各種方法を用いた予測誤差の比較と考察

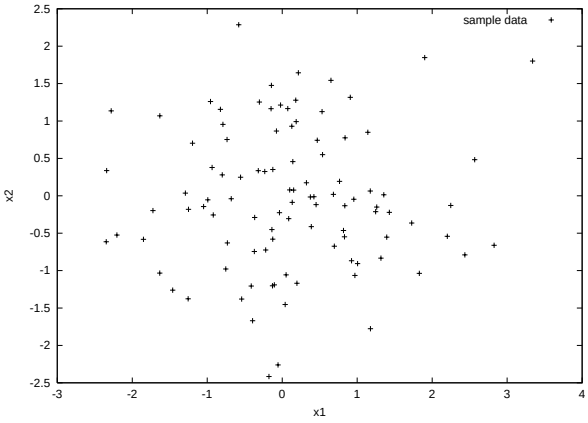


図 A.3.5 データの分布 (x_1-x_2)

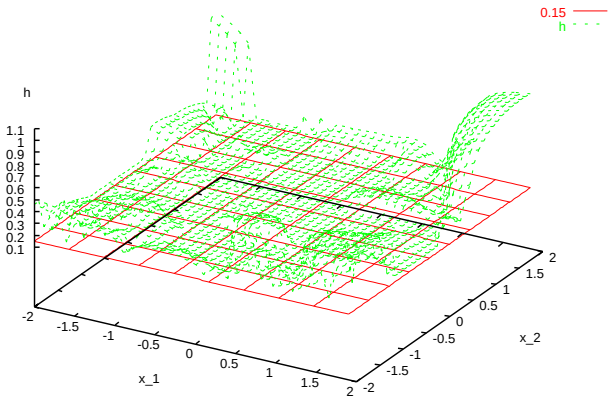


図 A.3.6 局所的な h の選択結果 (x_1-x_2-h)

表 A.3.1 は, x_1,x_2 が-2.5 から 2.4 まで 0.2 刻みの 25×25 格子点上での予測誤差を表す. ただし x_3 は 0 で固定した.

1)2) は, 距離関数の M の各対角成分 $m_{jj} = 1/|M|$ とし, 3)4) は, 式 (A.2.15) で表される評価関数を最小にする値,$m_1/|M| = 0.189, m_2/|M| = 0.982, m_3/|M| = 0.029$ を用いた. また 1)3) は重み関数の h を全域で 0.202 に固定し,2)3) は局所的に式 (A.2.13) が最小となる h を用いた予測結果である. ただし,h の最小値は 0.15 とした.

表 A.3.1 格子点上での予測誤差の比較

	M	h	平均	標準偏差	最大	最小
1)	fixed	fixed	0.0109	0.7357	1.8767	-5.0676
2)		opt	-0.0855	0.6644	1.2749	-4.0459
3)	opt	fixed	0.1315	0.4570	2.4140	-1.5906
4)		opt	0.1069	0.3616	1.7389	-1.4606

図 A.3.6 は，方法 2) における格子点上での最適な h の値を表す．図 A.3.5 で表されるデータ分布と比較すると，データが密な部分では h は最小値 0.15 となり，逆にデータが疎な範囲では h は大きな値を取っている事がわかる．

3)4) の予測誤差は 1)2) に比べて，平均は大きいが標準偏差は小さくなっており，全体としての予測誤差は小さくなっている．また，1) よりも 2)，3) よりも 4) の方が予測誤差が小さい事から，M を適切に設定し，h を局所的に適切に決定したほうが，予測精度が向上する事が予想される．

参考文献

[1] HONDA ASIMO
http://www.honda.co.jp/ASIMO/

[2] SONY QRIO
http://www.sony.co.jp/SonyInfo/QRIO/

[3] 有本 卓: “ロボティクスは先端技術になりうるか”, 日本ロボット学会誌, Vol.20 No.6, pp.569-570, 2002.

[4] R. A. Schmidt, & T. D. Lee: “Motor Control and Learning : A Behavioral Emphasis”, Human Kinetics, 1999

[5] Andersson: “A Robot Ping-Pong Player: Experiment in Real-Time Intelligent Control”, AT&T Bell Laboratories, The MIT Press, 1988.

[6] M. Ramanantsoa and A. Duray: “Towards a Stroke Construction Model,” International Journal of Table Tennis Sciences, No.2, pp.97-114, 1994.

[7] M. Bühler, D. E. Koditschek, and P. J. Kindlmann: “Planning and control of a juggling robot,” International Journal of Robotics Research, Vol.13, No.2, pp.101-118, 1994.

[8] J. L. Bentley: “Multidimensional Binary Search Trees Used for Associative Searching,” Communications of the ACM, Vol.18, No.9, pp.509-517, 1975.

[9] R. F. Sproull: “Refinements to Nearest-Neighbor Searching in k-d Trees,” Algorithmica 6, pp579-589, 1991.

[10] E. W. Aboaf, S. M. Drucker, and C. G. Atkeson: “Task-Level Robot Learning: Juggling a Tennis Ball More Accurately”, Proceeding of IEEE International Conference on Robotics and Automation, pp. 1290-1295, 1989.

[11] C. G. Atkeson, A. W. Moore, and S. Schaal: “Locally Weighted Learning,” Artificial Intelligence Review 11, pp.11-73, 1997.

[12] K. F. MacDorman: “Partition nets: An efficient on-line learning algorithm.”, Proc. of ICAR ’99: Ninth Int. Conf. on Advanced Robotics, Tokyo, pp.529-535, Oct. 1999.

[13] D. Gorinevsky and T. H. Connolly: “Comparison of some neural network and scattered data approximations: The inverse manipulator kinematics example”, Neural Computation, 6, pp.521-542, 1994.

[14] S. Arimoto, S. Kawamura, and F. Miyazaki: “Bettering Operation of Robots by Learning”, Journal of Robotic Systems, Vol. 1, No. 2, pp. 123-140, 1984.

[15] A.J.Ijspeert, J.Nakanishi, and S.Schaal: “Learning Attractor Landscapes for Learning Motor Primitives”, Advances in Neural Information Processing Systems 15 (NIPS2002), Becker S., Thrun S., Obermayer K. (Eds)

[16] Pierre Andry, Phillippe Gaussier, Sorin Moga, Jean Paul Banquet, and Jacqueline Nadel:, “Learning and Communicaiton via Imitation: An Autonomous Robot Perspective”, IEEE transactions on Systems Man and Cybernetics part A: systems and humans, Vol.31, No.5, pp.431-442, 2001.

[17] Yuichiro Yoshikawa, Minoru Asada, and Koh Hosoda: “Developmental Approach to Spatial Perception for Imitation Learning: Incremental Demonstrator’s View Recovery by Modular Neural Network”, IEEE Proceeding of RAS International Conference on Humanoid Robots, 2001.

[18] R. R. Burridge, A. A. Rizzi, and D. E. Koditschek: “Sequential Composition of Dynamically Dextrous Robot Behaviors,” International Journal of Robotics Research, Vol.18, No.6, pp.534-555, 1999.

[19] D. W. Aha and S. L. Salzbeerg: “Learning to Catch: Applying Nearest Neighbor Algorithms to Dynamic Control Tasks,” Proceeding of the Fourth International Workshop on Artificial Intelligence and Statistics, pp.363-368, 1993.

[20] S. Schaal and C. G. Atkeson: “Robot Juggling: An Implementation of Memory-Based Learning,” Control Systems Magazine, Vol.14, No.1, pp.57-71, 1994.

[21] 大築立志：「たくみ」の科学，朝倉書店，1988.

[22] H.Hashimoto, F.Ozaki, K.Asano, K.Osuka: “Development of Ping-Pong Robot System Using 7 Degree of Freedom Direct Drive Robots,” Proceeding of IEEE IECON 87 Industrial Application of Robotics and Machine Vision, pp.608-615,1987.

[23] F.Miyazaki, Y.Masutani, E.Hirose, D.Nakamura, and N.Sato: “State Estimating of a Spinning Ball Using LWR(Locally Weighted Regression),” Journal of Robotics Society of Japan, Vol.16, No.5, pp.108-113, 1998.

[24] T. Flash and N. Hogan: ”The Coordination of Arm Movements: An Experimentally Confirmed Mathematical Model”, J. of Neuroscience, Vol. 5, No. 7, pp. 1688 1703, 1985.

[25] 大村平：“予測のはなし”，日科技連出版.

[26] 森武：“卓球”, 西東社.

[27] Anne-Marie Brouwer, Eli Brenner and Jeroen B.J.Smeets: “Hitting Moving Objects –The dependency of hand velocity on the speed of the target–”, Exp. Brain Res., 133:242-248, 2000.

[28] R.J.Bootsma: “Timing an attacking fore-hand drive in table tennis”, J. of Exp. Psychol., Vol.16, pp.21-29, 1990.

[29] 宮崎文夫：“スキルと学習”，日本ロボット学会誌，Vol.13,No.1,pp.20-24,1995

[30] C. G. Atkeson, S. Schaal: “Memory-Based Neural Networks For Robot Learning”, http://www.cc.gatech.edu/fac/Chris.Atkeson

[31] Raymond H.Myers: “Classical and Modern Regression with Applications”

[32] Willam H.Press, Saul A.Teukolsky, William T.Vetterling ,and Brian P.Flannery: “Numerical Recipes in C(日本語版)”, 技術評論社.

[33] John J. Craig 著 三浦宏文・下山勲訳: “ロボティクス”, 共立出版株式会社, 1991.

[34] 安川電機編: “メカトロニクスのためのサーボ入門”，日刊工業，1986.

[35] F.Miyazaki, S.Kawamura, M.Matsumori, and S.Arimoto: “Learning contorol Scheme for a class of robot systems with elasticity”, Proceeding of the 25th conference of Desision and Contorol, Athens Greece, December 1986，pp.74-pp.79

関連文献

I. 学術論文集

[I1] Masahiro Takeuchi, Fumio Miyazaki, Michiya Matsushima, Masato Kawatani, and Takaaki Hashimoto "Dynamic Dexterity for the Performance of " Wall-Bouncing " Tasks", In *Proceedings of the International Conference on Robotics and Automation (ICRA2002)*, pp. 1559–1564, Washington D.C., USA, May 2002.

[I2] Fumio Miyazaki, Masahiro Takeuchi, Michiya Matsushima, Takamichi Kusano, and Takaaki Hashimoto "Realization of Table Tennis Task based on Virtual Target", In *Proceedings of the International Conference on Robotics and Automation (ICRA2002)*, pp. 3844–3849, Washington D.C., USA, May 2002.

[I3] 武内 將洋, 宮崎 文夫, 松嶋 道也, 河谷 雅人, 橋本 尚明"壁打ちタスクにおけるタスク実現の難易度の変化", 計測自動制御学会論文集, Vol.38, No.5, pp. 456–461, May 2002.

[I4] 宮崎 文夫, 武内 將洋, 松嶋 道也, 草野 貴充, 橋本 尚明"卓球タスクにおける仮想ターゲットの予測と実現方法", 日本ロボット学会誌, Vol.21, No.1, pp. 81–86, January 2003.

[I5] Michiya Matsushima, Takaaki Hashimoto, and Fumio Miyazaki "Learning to the Robot Table Tennis Task –Ball Control & Rally with a Human–", In *Proceedings of the International Conference on Systems, Man and Cybernetics (SMC2003)*, Washington D.C., USA, October 2003.

[I6] Fumio Miyazaki, Michiya Matsushima, Masahiro Takeuchi, and Takaaki Hashimoto "A Robot Plays Table Tennis: Ball Control and Rally with a Human Being", In *Proceedings of the International Conference on Methods and Models in Automation and Robotics (MMAR2004)*, Miedzyzdroje, Poland, August, 2004.

[I7] Michiya Matsushima, Takaaki Hashimoto, Masahiro Takeuchi, and Fumio Miyazaki: "A Learning Approach to Robotic Table Tennis", In *IEEE Transactions on Robotics*, 2005 (in print)

Ⅱ. 学術研究集会会議録

[Ⅱ1] 武内将洋，宮崎文夫，松嶋道也，河谷雅人，草野貴充: “誘導点を用いた卓球ラケットの制御”, 第 18 回日本ロボット学会学術講演会，pp. 819–820, 立命館大学, 2000.

[Ⅱ2] 橋本尚明，松嶋道也，草野貴充，武内将洋，宮崎文夫: “卓球タスクにおける仮想ターゲットの予測と実現”, 第 19 回 日本ロボット学会学術講演会, pp. 1085–1086, 東京大学, 2001.

[Ⅱ3] 武内将洋，河谷雅人，松嶋道也，宮崎文夫，橋本尚明: “壁打ちタスクにおける動的器用さ”, 第 19 回 日本ロボット学会学術講演会, pp. 1079–1080, 東京大学, 2001.

[Ⅱ4] 松嶋道也，橋本尚明，宮崎文夫: “メモリーベース学習を用いた卓球ロボットによるボール操作と対人ラリー”, 第 21 回 日本ロボット学会学術講演会, pp. 191, 東京工業大学, 2003.