

平成 2 1 年度戦略的基盤技術高度化支援事業

**「「アルゴリズム」からの、組込みシステム向け
「試作/製品用ソフトウェア」の実装に関する研究開発」**

研究開発成果等報告書

平成 2 2 年 7 月

委託者 東北経済産業局

委託先 東北イノベーションキャピタル株式会社

目 次

内容

第1章 研究開発の概要	2
1. 研究開発の背景・研究目的及び目標	2
(1) 研究開発の背景	2
(2) 研究の目的及び目標	3
2. 研究体制	8
(1) 管理体制	8
(2) 研究体制	8
(3) 委員会等	10
(4) 研究開発スケジュール	12
3. 成果概要	12
4. 当該研究開発の連絡窓口	13
第2章 本論	14
1. 研究の目的	14
2. 研究概要	14
3. 研究項目詳細	15
①アルゴリズムから試作/製品までの開発技術手法に係わる研究開発	15
②アルゴリズムモデルを基にした関数ライブラリの開発	19
③モデルコアの開発	24
④プロトタイプシステムの開発	28
第3章 全体総括	38
1. 研究開発成果	38
2. 今後の課題及び事業化展開	40
(1) 今後の課題	40
(2) 事業化計画	40
付録	42
1. 専門用語の解説	42
2. ライブラリー一覧	43
3. モデルコア一覧	45

第1章 研究開発の概要

1. 研究開発の背景・研究目的及び目標

(1) 研究開発の背景

本特定研究開発等計画は、中小企業の特定制品づくり基盤技術の高度化に関する指針(平成19年経済産業省告示第25号 平成20年度変更)に掲げる事項のうち、以下の項目に対応している。

(一) 組込みソフトウェアに係る技術に関する事項

1. 組込みソフトウェアに係る技術において達成すべき高度化目標

(1) 産業機械及び産業用ロボットに関する事項

①川下製造業者等の抱える課題及びニーズ

イ. 産業機械間のネットワーク化

エ. 新たな適合分野への対応

(4) 自動車に関する事項

①川下製造業者等の抱える課題及びニーズ

イ. 環境技術の高度化

エ. ITS の活用

(5) 川下分野横断的な共通の課題・ニーズに関する事項

①川下製造業者等の抱える課題及びニーズ

ア. コスト削減

ウ. 品質の向上

装置製品に求められる高度かつ斬新な機能を有するソフトウェアを開発する為には、ソフトウェア工学だけでなく、アプリケーション機能のアルゴリズム面での研究が必須である。例えば、本計画の協力者である東北大学青木孝文教授が研究されている「位相限定相関法」の応用例として、コンピュータビジョン分野、バイオメトリクス分野、画像センシング分野、工業用画像認識分野、波形解析・信号処理分野等における実用化が報告されている。日本の工業製品の国際競争力拡大の為には、これら研究成果の早期の実用化が必要である。

一方 これら大学への研究委託や研究組織における数値モデルベースの研究成果を、試作(プロトタイプ)製品(プロダクト)へと工程を進める際の課題として下記がある。

研究機関においては、*MATLAB* に代表されるアルゴリズム開発ツールを使用して組込みシステムに実装すべき「新機能」の研究を行っている。しかし、製品化に当たって検証が必要な「性能」や「必要とする資源(リソース)」「周辺ソフトウェアとの総合的な連携」を見積もる為には、実機に近い「プロトタイプ」を開発する必要がある。しかしながら前出のツールは、研究におけるトライアンドエラーを容易にするため 実行速度の遅いインタープリタ(逐次翻訳型)オブジェクトしか生成しない^{*1}。またプ

ロトタイプ化に必要となる C++等のコンパイル（一括翻訳型）オブジェクトを生成するためのプログラムのソースコードを生成するツールも存在するが、「アルゴリズム」を、組込みソフトウェアが必要とする「プログラムコード」に効率的に変換する事は現状困難であり、結果としてプロトタイプが必要とする「性能」の再現が難しい。そのため、川下企業では多くを手作業でプログラム作成をしており研究成果をプロトタイプによって検証する迄に多くの時間を要している。

※¹ 画像の類似性検知アルゴリズムの実行例として、*MATLAB* における実行時間が 26 秒必要な処理がある。本アルゴリズムを自動車の安全制御等のごく短時間での処理に利用すると仮定した場合、試作段階で処理時間を計測する必要がある。本例で、C++に変換した場合の実行時間は、0.066 秒となり製品化に向け有効な試作となった。（青木孝文教授）

プロトタイプからプロダクト（製品）化する際、上記で作成された組込みソフトウェアが、再利用を想定されていなかったり、粒度の細かい「関数ライブラリ」レベルに留まっている場合が多いため、プロトタイプ時に作成した組込みソフトウェアのプロダクト開発への再利用率が低く、早期製品化の障害となっている。

(2) 研究の目的及び目標

今回の研究開発の高度化目標は以下の通りである。

（１）産業機械及び産業用ロボットに関する事項

②高度化目標

イ．高性能化および機能の向上

（４）自動車に関する事項

②高度化目標

ア．信頼性の向上（機能安全確保を含む）

（５）川下分野横断的な共通の課題・ニーズに関する事項

②高度化目標

ア．生産性の向上（機能安全確保を含む）

今回の計画では上記課題を踏まえ、産業機械及び産業用ロボットや自動車を中心とした川下分野に共通して必要とされている、アルゴリズムから試作品や製品に実装される組込みソフトウェアを効率よく開発するための手法を開発する。

これにより、高度な機能を実現するための研究段階のアルゴリズムから、「試作」さらには「製品化」にいたる所要時間の短縮を図ることができるため産業機械及び産業用ロボットにおける高性能化および機能の向上、自動車における機能安全を実現する新たなアルゴリズムの開発から実装までの期間短縮が可能となる。

また、開発手法の実現にあたって、プロトタイプや製品のアプリケーションレベルで使用可能な抽象度の高いライブラリ関数を組み合わせたモデル化手法を取り入れる事により従来は、類似の機能アルゴリズムであっても「都度開発され使い捨て」される事が多かった試作段階で作成されたソフトウェアの再利用を促進することができるため、川下分野に共通して必要とされている、生産性の向上を図ることができる。

さらには、同手法を取り入れる事により、プログラム開発時に必要であった長年の経験を必要とするアルゴリズムの理解や組み込み装置固有の開発経験を抽象化したモデルとして扱うことができるようになるため、従来は大手企業のみで実施されていた、最先端のアルゴリズムからプログラム開発までの工程に、中小企業の参画が可能となり、地域産業振興や組み込みソフトウェア技術者の不足に資すると考えられる。

以上の高度化目標を実現するため、以下の方向性で研究開発を行う。

(一) 組み込みソフトウェアに係る技術に関する事項

2. 組み込みソフトウェア技術における高度化目標の達成に資する特定研究開発等の実施方法

(2) 開発技術の高度化（手法開発及びその支援ツールの開発）

①要求獲得・要求定義の高度化に関する研究開発

②システム設計・ソフトウェア設計に関する研究開発

③ソフトウェアの実装に関する研究開発

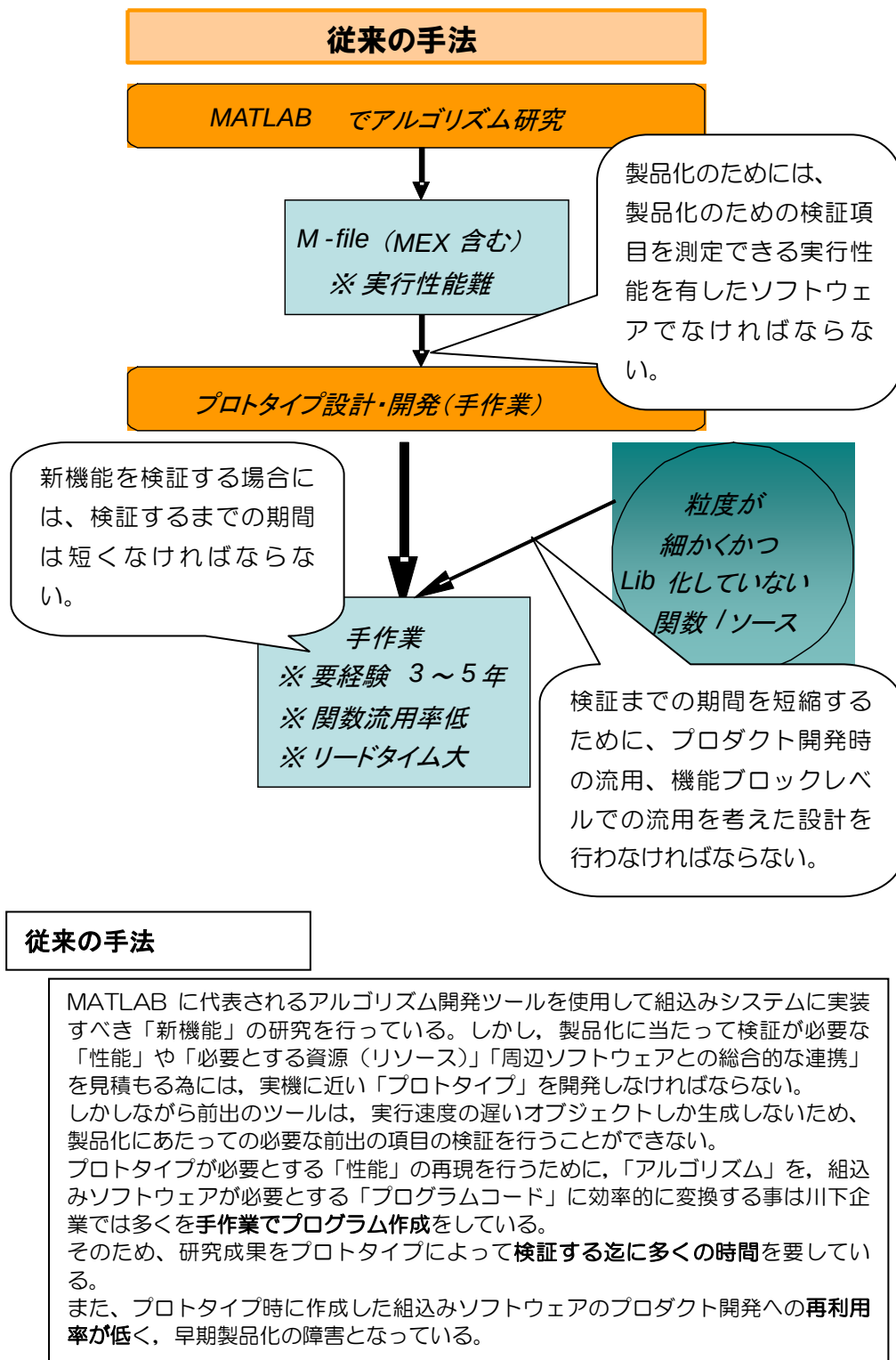
アルゴリズムを数値モデル化した「数値モデルプログラム^{※1}」から、組み込みシステムで高速に実行可能な「組み込みシステム向けプログラム」への効果的な変更手法を研究開発する。第一段階では、組み込みシステム向けプログラムを再利用可能な「関数」として実現する。上記の「関数」だけでは、機能単位が小さくなってしまい結果として非常に多くの関数が生成されることが予測される。また「関数」の意味するアルゴリズムの粒度も小さいため、これらを再利用する際に、アルゴリズムと関数の関係を熟知する人的経験が必要となる事が予想される^{※2}。

したがって、第二段として、「関数」をより粒度の大きな「モデル（モデルコア）」として再編成するとともに、モデルコアをより効率的に利用する為の、「モデル駆動型設計ツール」の適用を検討する。

※1 *MATLAB* では、*M-file* と呼ばれるインタプリタプログラムがこれに相当する。

※2 「関数」として「フーリエ変換」の機能があったとしても、これを「画像マッチング処理」といったアプリケーションとして実装するには、「位相限定相関法」という、より上位のアルゴリズムが必要で、上記を含む多くの関数の組み合わせが必要となる。

図. 従来手法のフロー



新たな手法

MATLAB でアルゴリズム研究

M-file (MEX 含む)
※ 実行性能難

プロトタイプ設計・開発 (MDD)

プロダクト開発 (MDD)

未整備の関数群を可視化し、ライブラリ化 (Lib) する

C++ MDD 用に関数 Lib 化

ModelCore

製品化までの期間を短縮

抽象度の高いライブラリ関数を組み合わせたモデル化モデル化

モデルコアを用いたプロダクト開発の短期化

新しい手法

日本のロボット・産業機械および情報家電のソフトウェア開発における国際競争力をこれまで以上に高めるためには、高度な機能を実現するための研究段階のアルゴリズムを、「プロトタイプ」さらには「プロダクト」にいたるまでの所要時間の短縮を図らねばならない。

そのために、新しい手法として、アルゴリズムの研究で作成される未整備の関数群を可視化し、機能関連毎に分類することによってライブラリ化 (Lib) するとともに、プロトタイプや製品のアプリケーションレベルで使用可能な抽象度の高いライブラリ関数を組み合わせたモデル化手法を研究開発する。このようにモデル形式で提供することにより、ターゲットに依存しないアーキテクチャの適用が可能となる。これをモデルコアと称する。

さらに、再利用を促進する手法として、本研究ではモデル駆動開発 (MDD) の適用にかかる効果を検証する。

本研究手法の効果

新しい手法を用いることにより、プロトタイプ開発時に「都度開発され使い捨て」となっていたソフトウェア関数群を抽象度の高いモデルとして管理できるため、再利用の促進効果が得られる。これにより、プロトタイプ開発からプロダクト開発へのソフトウェアの再利用により、研究開発で得られた要素技術をプロトタイプ開発からプロダクト開発までスムーズに適用可能となり、結果として工期短縮の効果も得られる。

また、機能ブロックレベルで抽象化されたモデルコアが資産化されることにより、最先端のアルゴリズムからプログラム開発までの工程に中小企業の参画が可能となり、地域産業振興や組込みソフトウェア技術者の不足解消の効果が期待できる。

また、本研究開発に関する研究開発項目及び技術的目標値は以下の通りである。

1. アルゴリズムから、試作(プロトタイプ)/製品(プロダクト)の設計・開発までの開発技術手法の研究開発
東北大学の協力を得て、従来の手法の問題点の分析を行い、新たな手法について具体的に実装可能な開発技術方法を検討し、基本設計書を完成させる。
2. アルゴリズムモデル(*M-File*)を基にした、関数ライブラリの開発
既に、大学に存在するプログラムソースコードを分析し、後工程での再利用が可能となるよう、関数ライブラリの開発を行う。
3. モデルコア (*ModelCore*) の開発
プログラムの再利用率を高めるため大きな機能単位で設計に利用可能な、「関数」を組み合わせた「モデル (モデルコア)」を開発する。
4. プロトタイプシステムの開発
アルゴリズムー関数ーライブラリーモデルコアを開発し、各フェーズにおける検証を行う。
本プロトタイプでは、工程時間について 30%程度の削減を目標とする。

2. 研究体制

(1) 管理体制

事業管理者

東北イノベーションキャピタル株式会社（最寄り駅：JR 仙台駅）

〒980-0014 宮城県仙台市青葉区本町 1-1-1 アジュール仙台 16F

（経理担当者） イバースタメント・マネージャー 長瀬 光信

（業務管理者） イバースタメント・マネージャー 竹井 智宏

管理員

氏 名	所属・役職	実施内容（番号）
竹井 智宏	イバースタメント・マネージャー	⑤

総括研究代表者（PL）

トライポッドワークス株式会社 代表取締役常務 菊池 務

副総括研究代表者（SPL）

トライポッドワークス株式会社 プロジェクトマネージャー 渋谷 義博

(2) 研究体制

トライポッドワークス株式会社

〒980-0811 宮城県仙台市青葉区一番町 1-1-41

カメイ仙台中央ビル 7F

氏 名	所属・役職	実施内容（番号）
菊池 務	常務取締役	①②③④⑤
山口 至	技術本部長	①②③⑤
渋谷 義博	マネージャー	①②③④⑤
山崎 政直	シニアマネージャー	①②③④
遠藤 一義	シニアマネージャー	①②③④
我妻 隆志	システムエンジニア	①②③④
梶原 道人	システムエンジニア	①②③④

庄司 豪	システムエンジニア	①②③④
須貝 恭一	システムエンジニア	①②③④
鳴海 康平	システムエンジニア	①②③④
小野寺 昭人	システムエンジニア	①②③④
斉尾 雅宏	システムエンジニア	①②③④
石井 勇樹	システムエンジニア	①②③④
山口 剛央	システムエンジニア	①②③④
李 始錫	システムエンジニア	①②③④

東杜シーテック株式会社

〒981-0913 宮城県仙台市青葉区昭和町 5-23

氏 名	所属・役職	実施内容（番号）
白川 清彦	マネージャー	①②③④
結城 一則	マネージャー	①②③④
塩谷 勝	チーフ	①②③④
藤江 英理子	システムエンジニア	①②③④
飯川 賢太	システムエンジニア	①②③④
鈴木 智浩	システムエンジニア	①②③④
村山 結美	システムエンジニア	①②③④
高橋 佐知子	システムエンジニア	①②③④
三浦 真実	システムエンジニア	①②③④
穴戸 利徳	システムエンジニア	①②③④
鈴木 直樹	システムエンジニア	①②③④
今野 範明	システムエンジニア	①②③④
山田 恵成	システムエンジニア	①②③④

バイスリープロジェクト株式会社

〒981-3212 宮城県仙台市泉区長命ヶ丘4丁目15番22号

氏 名	所属・役職	実施内容（番号）
千葉 和久	取締役	④⑤
樋下 祐司	チーフエンジニア	①②③④
一ノ瀬 友道	リーダーエンジニア	①④⑤
高橋 祐貴		①②③④
阿部 陽介		①②③④
鈴木 淳		①②③④
備前 克英		①②③④
須藤 哲也		①②③④

(3) 委員会等

研究開発推進委員会

氏 名	所属・役職	備考
菊池 務	トライポッドワークス株式会社 代表取締役 常務	PL
渋谷 義博	トライポッドワークス株式会社 プロジェク トマネージャー	SL
本田 光正	東杜シーテック株式会社 代表取締役社長	
白川 清彦	東杜シーテック株式会社 マネージャー	
結城 一則	東杜シーテック株式会社 マネージャー	
塩谷 勝	東杜シーテック株式会社 チーフ	
菅野 直	バイスリープロジェクト株式会社 代表取締 役社長	
一之瀬 友道	バイスリープロジェクト株式会社 技術部 リーダーエンジニア	
熊谷 巧	東北イノベーションキャピタル株式会社 代表取締役社長	
竹井 智宏	東北イノベーションキャピタル株式会社 インベストメント・マネージャー	

青木 孝文	東北大学 大学院情報科学研究科 教授	アドバイザー
畑岡 信夫	東北工業大学 教授 学科長	アドバイザー
佐々木 慶文	石巻専修大学 理工学部情報電子工学科 准教授	アドバイザー
門田 浩	独立行政法人情報処理推進機構ソフトウェアエンジニアリングセンタ 組込み系プロジェクトプロジェクトリーダー	アドバイザー
今井 和彦	宮城県産業技術総合センター	アドバイザー
太田 晋一	宮城県産業技術総合センター	アドバイザー

指導・協力

指導、協力者名	指導、協力事項
東北大学 大学院情報科学研究科 教授 青木 孝文	位相限定相関法、画像センシング、映像信号処理、バイオメトリクス
東北工業大学 教授 学科長 畑岡 信夫	組込みシステム、モバイル、ユビキタス、音声・画像認識
石巻専修大学 理工学部情報電子工学科 准教授 佐々木 慶文	信号処理、画像処理、ハードウェアアーキテクチャ
独立行政法人情報処理推進機構ソフトウェアエンジニアリングセンタ 組込み系プロジェクトプロジェクトリーダー 門田 浩	組込みソフトウェア利用技術、事業化プラン

(4) 研究開発スケジュール

実施内容	10月	11月	12月	1月	2月	3月	4月	5月	6月	7月
①アルゴリズムから試作/製品までの開発技術手法に係わる研究開発										
-1 従来手法における課題の抽出・整理		←→								
-2 基本設計		←→								
②アルゴリズムモデルを基にした関数ライブラリの開発										
-1 既存関数ライブラリの整理・分析	←→									
-2 関数ライブラリ開発ガイドラインの定義		←→								
-3 関数ライブラリ開発			←→							
③モデルコアの開発										
-1 既存プロトタイプ調査・分析	←→									
-2 モデルコア粒度・指針の定義		←→								
-3 共通インタフェースの設計			←→							
-4 モデルコアの開発				←→						
④プロトタイプシステムの開発										
-1 システム開発				←→						
-2 妥当性評価							←→			
⑤プロジェクトの管理・運営										
研究開発推進委員会の開催		○		○			○			○
報告書作成									←→	

3. 成果概要

第2章本論と重複する部分があるので、実施項目に対する成果のみを端的に示す。

①アルゴリズムから試作/製品までの開発技術手法に係わる研究開発

従来の「研究」「試作」「開発」に分離していた各フェーズを「モデル化」フェーズを追加し介することにより循環型サイクルを構築することを可能とした。本項目では、モデル化フェーズのプロセス設計を進め、「研究」から得られたソフトウェア資産を蓄積し、「試作」「開発」で利用かつフィードバックすることにより資産価値向上、ノウハウ蓄積が可能な循環型開発技術手法を構築した。

②アルゴリズムモデルを基にした関数ライブラリの開発

従来の研究や試作開発におけるソフトウェアは一度きりの使い捨てであった。本項目では当初から再利用を意識したライブラリ開発を行うことにより汎用 PC 向ライブラリから特定用途向ライブラリまで合計4プラットフォーム向け、110関数の開発を行った。ベンチマーク性能測定を実施し現在市場で使用されているライブラリとの性能比較を行うとともに、実際にプロトタイプ開発で利用し精度などの適用度を検証するなどライブラリ開発におけるノウハウを蓄積した。

③モデルコアの開発

これまでの試作ではアプリケーションとライブラリの境界が曖昧で明確化されていなかった。本項目ではあらかじめ再利用を目的にアプリケーションとライブラリの境界を明確にし、アプリケーション部をモデル駆動開発手法を採用して開発することによりソフトウェアの可視化、再利用性の向上を達成することが出来た。また、ツールと連動することによりコード自動生成やマルチプラットフォーム化が可能となり使用性の向上を実現した。

④プロトタイプシステムの開発

プロトタイプシステムの開発では、本研究開発で開発したソフトウェア資産（関数ライブラリ、モデルコア）を今回設定した3つの標準プラットフォームに適用して実際に組み込んで開発を行ったうえで妥当性検証を行った。その結果、本開発において約8割の再利用率を確認した。また、再利用するシステムによっては主要構成をそのまま再利用することも可能で、そのため不具合発生率も非常に低く、本研究開発におけるソフトウェア資産の開発手法および再利用技術の有効性を証明できたといえる。

4. 当該研究開発の連絡窓口

トライポッドワークス株式会社 プロジェクトマネージャ 渋谷 義博
Tel:022-227-5680 FAX:022-227-5685
e-mail : y.shibuya@tripodw.jp

第2章 本論

1. 研究の目的

各産業分野における組込みソフト開発の国際競争力をこれまで以上に高めるためには、高度な研究段階のアルゴリズムから実際の組込みシステムで検証するまでの所要時間の短縮化を図る必要がある。

本研究開発では、研究過程で作成される未整備の関数群を可視化して機能関連毎にライブラリ化し、さらにプロトタイプや製品レベルで利用可能な抽象度の高いモデル（以下モデルコア）を用いた開発技術手法を確立し、工期の短縮を図るものである。また、機能ブロックレベルで抽出されたアプリケーションをモデルコアとして資産化することで、最先端のアルゴリズムからプログラム開発の工程に中小企業の参画を可能にする。

2. 研究概要

産業機械及び産業用ロボットや自動車を中心とした川下分野に共通して必要とされている、アルゴリズムから試作品や製品に実装される組込みソフトウェアを効率よく開発するための手法を開発する。

これにより、高度な機能を実現するための研究段階のアルゴリズムから、「試作」さらには「製品化」にいたる所要時間の短縮を図ることができるため産業機械及び産業用ロボットにおける高性能化および機能の向上、自動車における機能安全を実現する新たなアルゴリズムの開発から実装までの期間短縮が可能となる。

また、開発手法の実現にあたって、プロトタイプや製品のアプリケーションレベルで使用可能な抽象度の高いライブラリ関数を組み合わせたモデル化手法を取り入れる事により、従来は類似の機能アルゴリズムであっても「都度開発され使い捨て」される事が多かった試作段階で作成された組込みソフトウェアの再利用を促進することができるため、川下分野に共通して必要とされている、生産性の向上を図ることができる。

さらには、同手法を取り入れる事により、プログラム開発時に必要であった長年の経験を必要とするアルゴリズムの理解や組込み装置固有の開発経験を抽象化したモデルとして扱うことができるようになるため、従来は大手企業のみで実施されていた、最先端のアルゴリズムからプログラム開発までの工程に、中小企業の参画が可能となり、地域産業振興や組込みソフトウェア技術者の不足に資すると考えられる。

本事業では、アルゴリズムから試作/製品までの開発技術手法に係わる調査と手法の確立を行う。また、既に大学に存在する資産を関数ライブラリやモデルコアに再利用可能な形に整備するとともに、実際にプロトタイプ開発を進めながら本開発技術手法における妥当性検証を実施する。

3. 研究項目詳細

①アルゴリズムから試作/製品までの開発技術手法に係わる研究開発

<実施内容>

研究から試作（プロトタイプ）さらには製品（プロダクト）の設計・開発までの一貫した開発技術手法を研究する

<目標>

研究開発から試作・製品開発までの一貫した再利用性向上のための開発技術手法を考案し基本設計書にまとめる

<達成状況>

①-1 従来手法における課題の抽出・整理

組込み産業におけるモデル開発手法は、自動車業界に代表される制御モデルを用いた手法が主流である。制御モデルは、制御対象（プラント）に対する制御装置（コントローラ）を定義するものであり、制御ブロックモデリング（構造化モデリング）と呼ばれる。主に *MATLAB* や *LabVIEW* などのツールが利用されている。しかし、本モデルは上流工程設計、シミュレーションを目的とした特定分野向けのモデリングであり、本研究開発が目標とするシステム、アプリケーションの全体振る舞いを定義する開発効率を向上させるモデル開発とは異なる。本研究開発では、ソフトウェアの可視化や自動コード生成などの分野を特定せずに（プラットフォームを意識せずに）適用できことを考え *UML* によるオブジェクト指向モデリングを採用する。

基本設計を行うにあたり *UML* 対応のモデリングツールの調査を行った。今回 *UML* を採用した理由として、*UML* はモデル記述言語のデファクトスタンダードとなっており、*UML* で記述されたデータは、ツールに依存せずに永続的に再利用可能となり将来におけるツール発展のメリットを享受できるものと判断した（本研究開発においては、ツール類は補完関係と位置付けて分離しておりツールにおける機能を定義することは本研究開発では行わない。しかし、より現実的な成果を残すためにも現在最適なツールを選択し取り入れることとした）。そのことから、標準互換性と開発効率の観点から、以下の点を中心に調査を行った。

- ✓ *UML* の記述が可能か（対応バージョンは）
- ✓ インポート／エクスポート機能があるか
- ✓ 自動コード生成が可能か
- ✓ リバースエンジニアリングが可能か
- ✓ マルチプラットフォームに対応しているか

現在市場に投入されている有償／無償の計9製品について調査を行った。その結果、*UML* 記述が可能な製品が6製品、そのうちインポート／エク

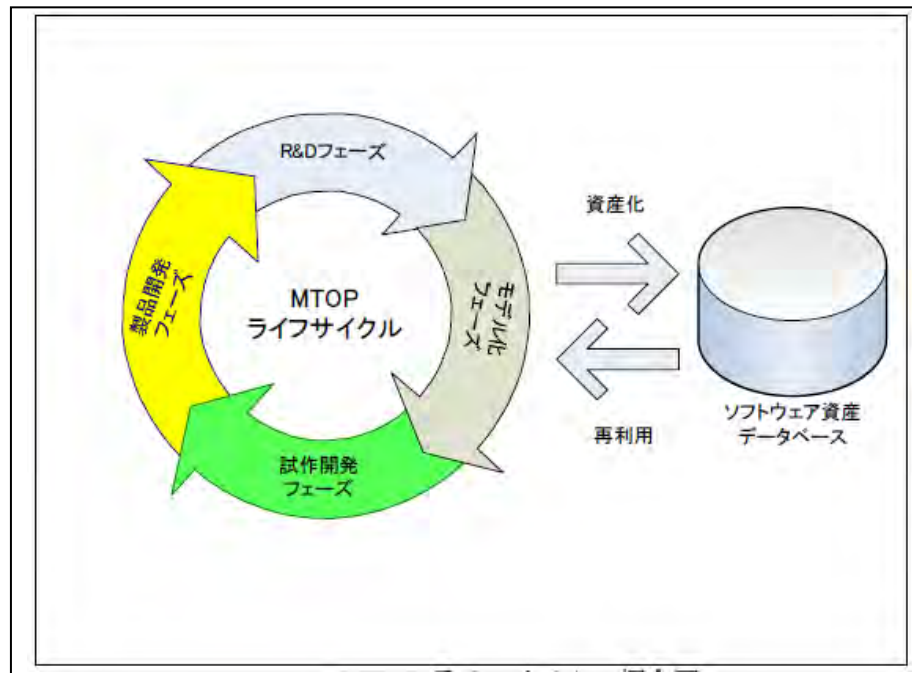
サポート、自動コード生成、マルチプラットフォーム対応までを考慮に入れると「IBM® Rational® Rhapsody®」が現時点では最適であるとの結論を得た。

①-2 基本設計

本開発技術手法の研究は、研究機関である東北大学が有する研究資産をシームレスに産業界にシーズアウトすることを目的とし、高度な数値計算ライブラリを部品化し、かつアプリケーションのフレームワークもモデルコアとして部品化することにより再利用性を高めようとするものである。

本研究を行うにあたり、これまで別々の組織・プロセスで行われてきた「研究」「プロトタイプ開発」「製品開発」などのフェーズをひとつのライフサイクルとして定義することから始め、そこに部品化プロセスを混入することによる全体プロセス設計を行った。

図. *MTOP* ライフサイクル概念図



部品化するフェーズを「モデル化フェーズ」と定義しそこに必要とされかつ研究や他開発フェーズに対し疎結合ながら循環可能なライフサイクルを定義することが重要となる。

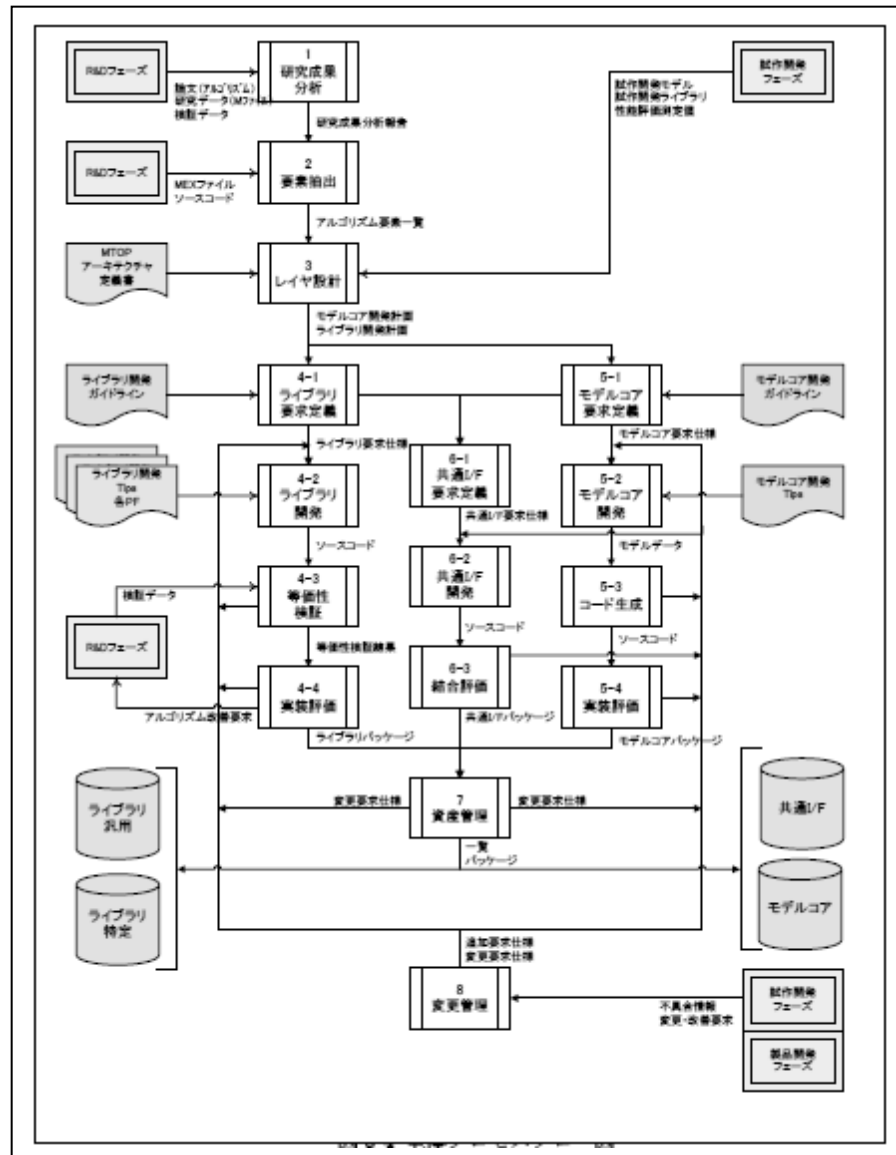
以下は基本方針と全体プロセスフローである。

<基本方針>

- ✓ 研究資産をシームレスに活用可能なこと
- ✓ アーキテクチャ方針に基づいた開発プロセスであること
- ✓ 研究成果との等価性および資産性能を可視化すること

- ✓ フィードバックプロセスを持った循環型プロセスであること

図. モデル化フェーズ全体プロセスフロー



全体プロセスにおける定義ドキュメント、各プロセスでの入出力ドキュメント・データおよび各プロセスで対処すべき作業内容についてガイドラインを定義した。本フェーズによりライフサイクルを通して効率的な部品化と永続的な保守が可能となり、本研究にて目標とする一貫した技術開発手法を確立した。

＜課題および対策＞

モデル化フェーズを担う部品化部隊は研究や開発部隊とは別に存在し、並行的に強調しながら作業をすすめることが大事である。また部品化は開発のフォローとして能動的に動けることが求められる。しかし、今回研究開発されたソフトウェア資産を事業終了後に開発および保守・フォローする体制が必要である。今後は本事業における研究開発企業にて共同で対応を

行いながら、開発コンソーシアムを起こして会員企業による自主運営の道を固める必要がある。

＜研究開発成果＞

- ・ 開発技術手法 基本設計書
- ・ *MTOP* アーキテクチャ定義書

②アルゴリズムモデルを基にした関数ライブラリの開発

＜実施内容＞

大学に存在する位相限定相関法(以下 *POC*)に関連した未整備の関数群を可視化して機能関連毎に体系化し *POC* 関数ライブラリを構築する

＜目標＞

関数ライブラリを開発するうえでの開発フローを標準化してガイドライン化するとともにプロトタイプシステム開発で行うシステムに対応した関数ライブラリを開発を行う

＜達成状況＞

②-1 既存関数ライブラリの整理・分析

大学に存在する *POC* を用いたプロトタイプシステムの調査・解析を行い、高度な数値計算を行うコア関数や、それらを組合せた必要な機能関数群の整理を実施。複数のプラットフォームを対象としたライブラリ開発を目的として、調査・解析の対象としたプロトタイプシステムは以下とした。

表. プロトタイプシステム一覧

プロトタイプシステム	対象プラットフォーム
虹彩認証システム	<i>DSP</i>
3D 顔復元システム	<i>Windows</i>
リアルタイム 3D 顔復元システム	<i>GPU</i>

また、大学に存在するプロトタイプシステム以外にも、既存画像処理アルゴリズムやオープンソース (*OpenCV*) の調査・解析を実施し、画像処理ライブラリとして必要な処理の整理を実施した。

主なアルゴリズムとしては、*SIFT*、*SURF*、*HOG* が挙げられる。

②-2 関数ライブラリ開発ガイドラインの定義

関数ライブラリ開発の標準化を目的として、ガイドラインの作成を実施。ガイドライン作成にあたっては、ライブラリ開発の位置付け、開発フロー、各工程における成果物や作業手順を明確にした。これにより本研究開発の目的である関数ライブラリ開発の標準化、作業効率の向上に繋がると考えられる。

また、「組み込みソフトウェア開発向けコーディング作法ガイド[C言語版/」をもとに、コーディング規約の作成を実施。これにより、関数ライブラリ開発時の可読性、保守性、品質の向上に繋がると考えられる。

②-3 関数ライブラリ開発

②-1 で整理した関数群の中から関数ライブラリを抽出し、プロトタイプ

開発に必要な各プラットフォームに対応した関数ライブラリの実装を開発を実施。

本研究開発で開発するライブラリのカテゴリについては、*OpenCV* を参考に、以下とした。なお、他にも *OpenCV* にはカテゴリが存在するが、本研究開発で開発するプロトタイプシステムをターゲットとして、必要最小限のカテゴリとしている。

表. 関数ライブラリのカテゴリ分類

No.	カテゴリ名	No.	カテゴリ名
1	エッジ検出	9	物体検出
2	特徴点検出	10	キャリブレーション
3	サンプリング	11	信号処理
4	画像変換（幾何変換）	12	相関関数
5	フィルタ	13	正弦波
6	色変換	14	配列操作
7	輪郭検出	15	補間
8	マッチング	16	初期化

※開発した関数ライブラリについて「添付 ライブラリー一覧」を参照

各プラットフォーム向けのライブラリは、「汎用 CPU 向け、標準ライブラリ」、「GPU 向け GPU 版ライブラリ」、「DSP 向け DSP 版ライブラリ」、「組み込みライブラリ」の4ライブラリとし、標準ライブラリは、プラットフォームに依存しないライブラリ群で構成されており、組み込みシステムから Windows に至るまで利用可能としている。GPU 版ライブラリについては、C 言語開発環境が整備されている CUDA (NVIDIA 社製) を対応プラットフォームとし、DSP 版については、携帯端末等に広く採用されている OMAP (TI 社製) を対応プラットフォームとした。

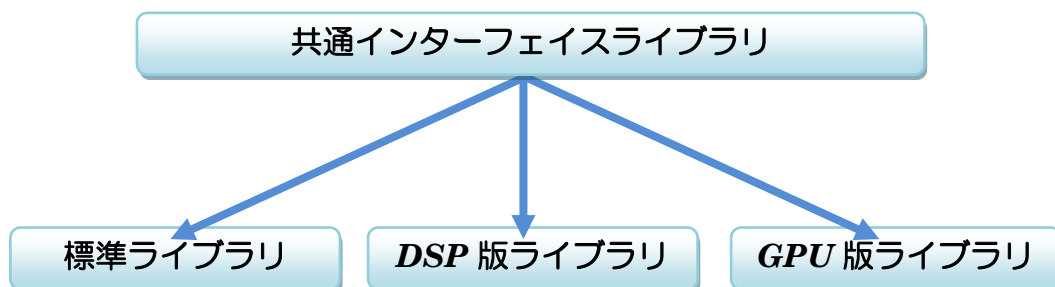
表. 対象プラットフォーム

ライブラリ種別	対象プラットフォーム	識別子
標準ライブラリ	汎用 CPU	<i>Image Process</i>
GPU 版ライブラリ	CUDA (NVIDIA)	CUDA
DSP 版ライブラリ	OMAP (TI)	DSP
組み込みライブラリ	TriBOARD	EMB

各プラットフォームへの適応手段は、当初変換テーブル等による対応を検討したがユーザーインターフェイスの構築や各関数ライブラリを汎用的に開発する必要があることから、関数ライブラリのボリュームが増大することと、それに伴う処理速度の問題が考えられるため、プラットフォーム共通のインターフェイスライブラリと、各プラットフォームに

対応した個別の関数ライブラリの構造とした。これにより、アプリケーションは、プラットフォームを意識する必要なくライブラリを使用することが可能である。

図. 共通インターフェース概念図



本研究開発で開発したライブラリの性能測定については、*OpenCV* を比較対象として同じ機能を有する以下の標準ライブラリの処理速度を計測し、比較・検証を行った。

[測定条件：標準ライブラリ、*OpenCV*]
 ハードウェア： *Intel Core2Duo 2.26GHz*
 メモリ : *2.9GB*

[測定条件：GPU ライブラリ]
 ハードウェア： *Intel Core i7 Extreme 2.22GHz*
 メモリ : *6GB*
 GPU : *NVIDIA GeForce GTX275*

表. 性能測定結果

画像サイズ変換	<i>Min</i>	<i>Ave</i>	<i>Max</i>
標準ライブラリ	<i>11.54msec</i>	<i>11.63msec</i>	<i>11.86msec</i>
<i>CUDA</i> 版	<i>2.76msec</i>	<i>3.01msec</i>	<i>3.42msec</i>
<i>OpenCV</i>	<i>0.95msec</i>	<i>0.98msec</i>	<i>1.02msec</i>

平滑化	<i>Min</i>	<i>Ave</i>	<i>Max</i>
標準ライブラリ	<i>172.45msec</i>	<i>191.22msec</i>	<i>200.37msec</i>
<i>CUDA</i> 版	<i>7.21msec</i>	<i>7.92msec</i>	<i>9.32msec</i>
<i>OpenCV</i>	<i>2.29msec</i>	<i>2.35msec</i>	<i>2.53msec</i>

グレースケール変換	<i>Min</i>	<i>Ave</i>	<i>Max</i>
標準ライブラリ	<i>5.97msec</i>	<i>6.06msec</i>	<i>6.32msec</i>
<i>CUDA</i> 版	<i>2.48msec</i>	<i>2.52msec</i>	<i>2.56msec</i>
<i>OpenCV</i>	<i>0.79msec</i>	<i>0.81msec</i>	<i>0.87msec</i>

※性能測定に使用した画像サイズは 320×240

上記測定と同様の画像サイズを使用し、*ARM* プロセッサ用にビルドした標準ライブラリと組み込みライブラリの測定結果は次のとおりである。

[測定条件]

ハードウェア：*S3c2440A* (*ARM9* コア 400Mhz) 搭載評価ボード

OS：*Windows Emebedded CE6.0*

測定方法：処理時間はミリ秒単位で取得 (*GetTickCount* 関数を使用)。

GetTickCount 関数 分解能：1msec 秒、誤差：1～11msec

表. 組み込みにおける性能測定結果

画像サイズ変換	<i>Min</i>	<i>Ave</i>	<i>Max</i>
標準ライブラリ	427 msec	432 msec	439 msec
組み込みライブラリ	48 msec	48 msec	49 msec

モルフォロジー演算 (<i>Diration</i>)	<i>Min</i>	<i>Ave</i>	<i>Max</i>
標準ライブラリ	4212 msec	4256 msec	4281 msec
組み込みライブラリ	365 msec	368 msec	374 msec

グレースケール変換	<i>Min</i>	<i>Ave</i>	<i>Max</i>
標準ライブラリ	1701 msec	1701 msec	1701 msec
組み込みライブラリ	34 msec	35 msec	35 msec

※性能測定に使用した画像サイズは 320×240

標準ライブラリの測定結果は、*OpenCV* と比較すると処理速度が劣るが、*OpenCV* はプラットフォームを意識して *CPU* やメモリを効率よく使用しているのに対し、開発した標準ライブラリは、プラットフォームを意識せず開発していることと、エラー処理の充実を図ったことが要因として考えられる。

また、組み込みプラットフォームにおける測定結果を比較すると、標準ライブラリに比べ組み込みライブラリが10倍以上処理速度が速いという結果を得た。これにより、組み込みライブラリ内で行っている固定小数点化の効果が大きい。ただし、固定小数点化を行った場合は、処理速度と演算精度とのトレードオフとなるため、注意が必要である。

なお、高度な数値演算を高速に処理させるため開発した *GPU* 版ライブラリは、標準ライブラリと比較すると 2.4 倍程の高速化が図られている。ただし、標準ライブラリ、*GPU* 版ライブラリともに高速化の余地があり、今後の課題ともいえる。

<課題および対策>

今後の課題として以下が挙げられる。

- ・ 関数ライブラリの充実とバリエーションの拡大
今回分類したカテゴリや、カテゴリ毎に定義した関数ライブラリでは、まだ不足しているライブラリがあると考えられる。そのため、今後さらなるライブラリ群の充実を図る必要があると考えられる。また、対応プラットフォームのバリエーションの拡大、より汎用的なライブラリとしてのオープン規格（*OpenCL* 等）への対応も必要と考える。
- ・ 関数ライブラリ的高速化
本研究開発で開発したライブラリは、*OpenCV* と比較すると処理速度が劣っている。メモリ使用の効率化やエラー処理を改善することにより処理の高速化を図る必要があると考える。

<研究開発成果>

- ・ コーディング規約
- ・ 関数ライブラリ開発ガイドライン
- ・ 共通関数説明書
- ・ 開発 *Tips* 集
- ・ 関数ライブラリ *Spec* シート
- ・ プラットフォーム適合表
- ・ ライブラリパッケージ

③モデルコアの開発

<実施内容>

ライブラリから抽象度を高めたアプリケーションレベルで再利用可能な「モデルコア」を開発する

<目標>

ハードウェア依存の関数ライブラリと分離したハードウェア非依存の汎用モデルとなるモデルコアを開発し、今後の試作・製品開発におけるアプリケーションで再利用可能な部品とする

<達成状況>

③-1 既存プロトタイプの調査・分析

東北大学の研究資産である掌紋認証アルゴリズムを用いた認証システムを対象とし、アプリケーションの構造解析と関数ライブラリの分離を実施した。

本分析により、掌紋アルゴリズムで利用している画像処理機能と、*UI* をトリガーとした画面遷移や、システムの状態遷移を担う箇所を分離した。この分類方法はすべての研究成果に適用でき、研究成果から *MTOP* 資産に落とし込むうえでの効果的な方法である。

③-2 モデルコア粒度・指針の定義

モデルコアとは、ソフトウェアの構成（静的な設計）と振る舞い（動的な設計）を併せもったものとして定義する。更に、このモデルコアはモデル駆動開発手法（以下、*MDD* 手法）による開発を採用し、*C++* 言語で記載された処理を記載した実行可能なモデルとして定義した。なお、モデルは *UML2.0* に準拠し記述を行っている。

プラットフォームに非依存となるモデルコアを開発するに当たり、モデルコアを構成する要素を以下のように定義した。

- ✓ 基本となるモデルコア
- ✓ システム要件によって変更しなければならない部分（拡張部分）
- ✓ 外部 *UI* とのインタフェース部分（ソフトウェアプラットフォーム依存部分）
- ✓ 画像処理ライブラリ呼び出し部分（ハードウェアプラットフォーム依存部分）

これによりプラットフォーム非依存のモデルコア開発を達成している。

また、この分類により、基本となるモデルコアの類似システムへの再利用や、別プラットフォームへの同一システムの流用といった、アプリケーションの再利用性の向上を容易にすることができた。

それに加え、*UML* を利用したことにより、オブジェクト指向ソフトウェア

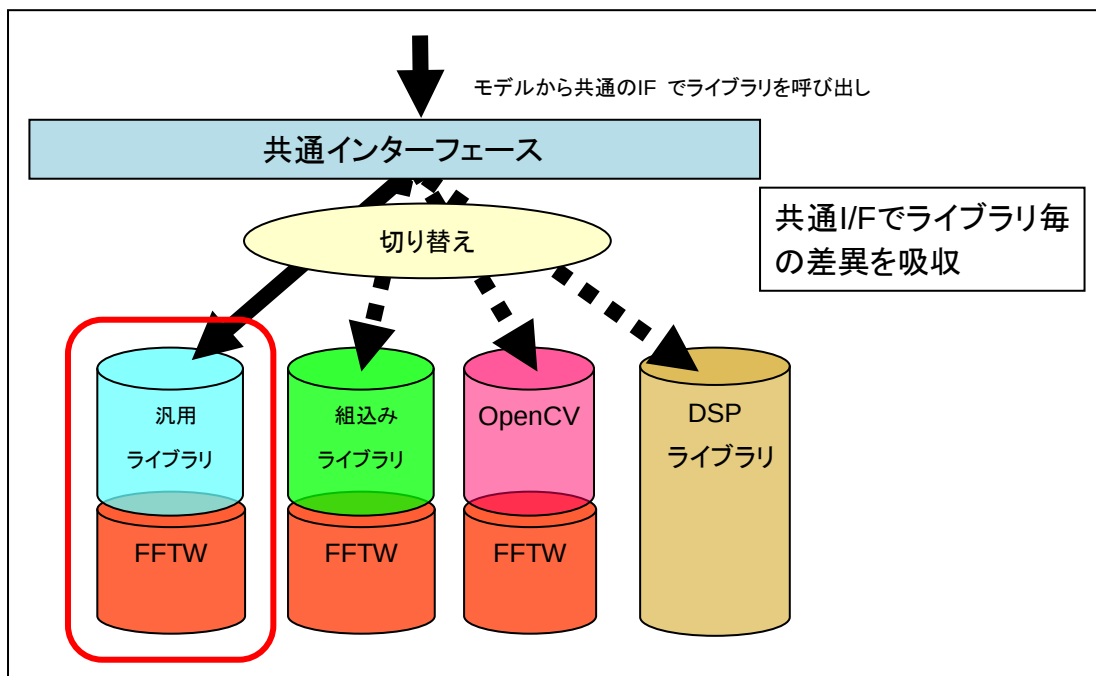
ア設計として一般的なクラス／パッケージ／デザインパターンの利用により、再利用性の向上目標が達成できた。

③-3 共通インタフェースの設計

ハードウェアプラットフォーム毎のライブラリの差異を吸収するために、共通インタフェースを用意する。この共通インタフェースの *API* をモデルコアから呼び出すことによって、個々のライブラリの差異をモデルコアが意識することなく、ハードウェアプラットフォーム毎のライブラリを用意することが可能となる。

共通インタフェースとしては、C 言語向けライブラリと、C++言語向けクラスライブラリの 2 種類を提供する。提供する機能はどちらも同じであり、開発者にとって使いやすいインタフェースでライブラリを利用できるようにした。

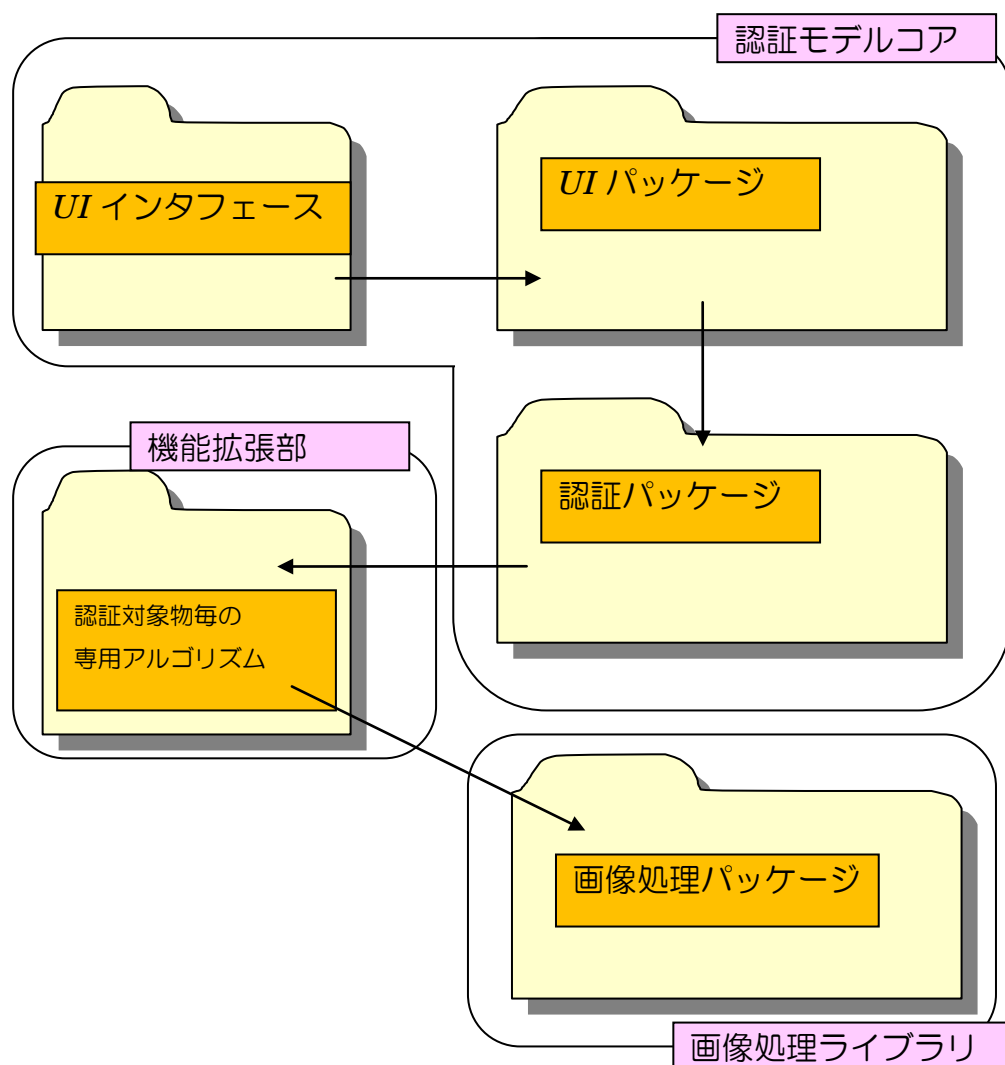
<共通インタフェース概念図>



③-4 モデルコアの開発

定義したモデルコア粒度・指針にのっとり、認証モデルコアを開発した。開発したモデルコアを次図に示す。

<認証モデルコア構成図>



- ・ 認証モデルコア：
認証システムのコア部分。認証システムとしての振る舞いを司る部分。
この部分を利用することによって、物体を認証するシステムの振る舞いを再利用可能とする。外部 *UI* とのインタフェース部分は、*UI* インタフェースとし、外部 *UI* で実装することによって、認証モデルコアに処理を要求することを可能とする。
- ・ 機能拡張部：
認証の対象物によって、カスタマイズを行う部分。物体や人物、静脈、掌紋、虹彩など、対象物毎の最先端の研究アルゴリズムを実装することにより、様々な派生システムへの再利用が可能となる。
- ・ 画像処理ライブラリ：
共通インタフェース呼び出し部分。特定ハードウェア用のライブラリを

呼び出す場合も、共通インタフェースでライブラリの差異を吸収するため、ハードウェアプラットフォーム非依存でシステムを開発することが可能となる。

認証モデルコアを利用し、掌認証モデル、虹彩認証モデルを開発した。各モデルの有効ステップ数を下表に示す。また、掌認証モデル虹彩認証モデルにおいては、認証モデルコアを基準とした新規開発ステップ数、再利用ステップ数、再利用率を記載する。

表. 認証モデルをベースとしたモデルの Step 数比較

	認証モデルコア	掌認証モデル	虹彩認証モデル(※)
有効ステップ数	2894 Step	3757 Step	3074 Step
新規開発ステップ数	2894 Step	863 Step	180 Step
再利用ステップ数	—	2894 Step	2894 Step
再利用率	—	77.0 %	94.1 %

※ 虹彩認証モデルにおいて、一部、掌認証モデル内で実装している処理を UI 部で実装するよう設計を行っているため、モデルコアの新規開発ステップ数が、掌認証モデルの新規開発ステップ数に比べ少ない。ただし、認証モデルコアに機能追加して虹彩認証モデルを作成しているため、再利用ステップ数は認証モデルコアのステップ数と等しい。

上表により、派生システム開発におけるソフトウェア再利用率は 77%と高く、再利用率の高いソフトウェアの開発を達成したといえる。

<今後の課題および対策>

今後のモデルコアを利用していく上での課題として、次の2点がある。

- ・ 共通インタフェースにおけるライブラリの組み合わせ
ハードウェアプラットフォームの独立性を高めるために、共通インタフェースという機能を用意している。この共通インターフェースは現時点では汎用ライブラリ/組込みライブラリ/DSP ライブラリ/GPU ライブラリをサポートしているが、任意の組み合わせでライブラリを使用することができない。すなわち、A という機能は、組込みライブラリのものを使用し、B という機能は DSP ライブラリを使用することができない。今後、モデルコアを利用したプロトタイプ開発の柔軟性を高めていく上では、柔軟にライブラリを組み合わせることへの要求が予想されるため、共通インタフェースとして、*configure* のようなライブラリを半自動的に組み合わせるような仕組みを検討していく。
- ・ 開発ツールを利用したライブラリの組み合わせ
今回のモデルコアは開発ツールとして、IBM® Rational® Rhapsody® (以下、*Rhapsody*) を利用している。今後、*Rhapsody* 以外の開発ツールを利用する場合、*Rhapsody* から XMI 形式でエクスポートすることによって、*Rhapsody* 以外のツールにインポートすることが可能となる。今後、別のツールで開発を行う場合は、XMI 形式でインポート可能であ

り、モデルらの自動生成が可能であることがツール選定における条件となる。

<研究開発成果>

- ・ モデルコア一覧
- ・ モデルコア開発ガイドライン
- ・ モデルコアモデリング規約
- ・ モデルコア開発 *Tips* 集
- ・ モデルコアパッケージ

④プロトタイプシステムの開発

<実施内容>

本開発技術手法の適合度を検証する為に具体的な開発環境を用いたプロトタイプの開発を行う

<目標>

本開発を通して研究アルゴリズムから関数ライブラリ、モデルコア、アプリケーションの一連の開発プロセスを通した妥当性の検証を行う

<達成状況>

④-1 システム開発

システム開発では、本開発技術手法に従って標準プラットフォームとして定義した3プラットフォームのプロトタイプシステムの開発を行い、開発ノウハウの蓄積・フィードバックを行った。

表. プロトタイプシステム一覧

標準プラットフォーム	プロトタイプシステム
携帯プラットフォーム	掌認証システム
	虹彩認証システム
PC/GPU プラットフォーム	3D 顔復元/リアルタイム 3D 顔復元システム
FA プラットフォーム	3D ピッキングシステム

1. 掌認証システム

掌認証モデルコアと画像処理ライブラリを利用した掌認証システムのデスクトップアプリケーションの開発を実施した。また、モデルコアとライブラリの組み合わせによって、様々なプラットフォームへ対応可能であることを検証するため、掌認証モデルを再利用し画像処理ライブラリを組み込み用のライブラリと組み合わせ、組み込み用アプリケーションの開発を実施した。

下図に、デスクトップアプリケーションから、組み込み用アプリケーションプロトタイプを派生開発する場合の工程図と、開発したアプリケー

ションの UI 画面を示す。

図. 掌認証システムにおけるプラットフォーム変更時の作業フロー図

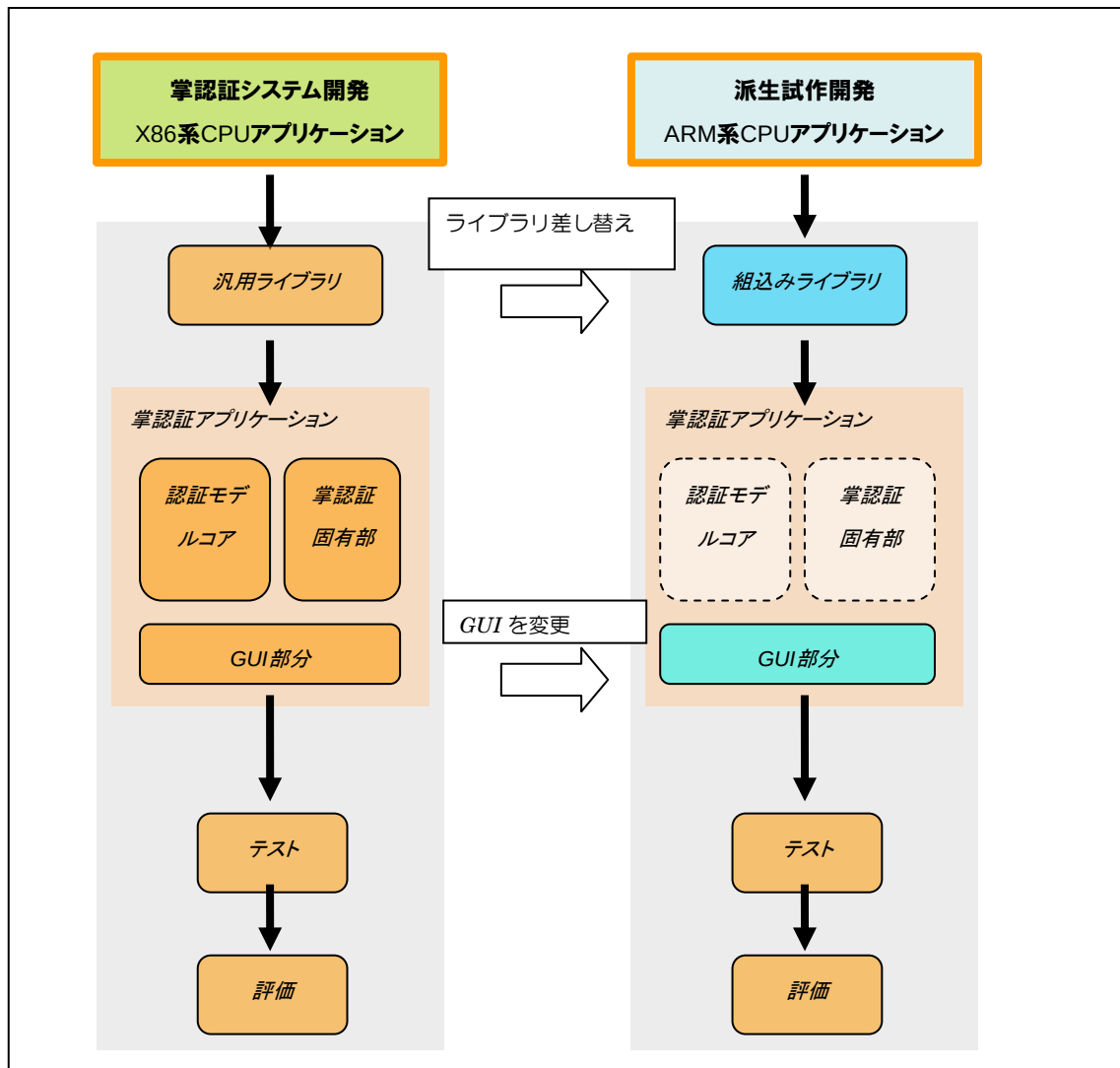


図. 掌認証モデルを利用して開発した掌認証アプリケーション



2. 虹彩認証システム

標準ライブラリの妥当性検証のため、PC 版の掌紋認証で開発したモデルコアをベースとして虹彩認証のモデルを開発し、本研究開発で開発した標準プラットフォーム向けの関数ライブラリを用いて PC 版の虹彩認証システムのプロトタイプシステムの開発を行った。

図. PC 版虹彩認証システム



また、OMAP を対象として開発した DSP 版ライブラリの妥当性検証のため、PC 版の虹彩認証システムを流用し、DSP 向けの関数ライブラリを用いて OMAP 上で動作する虹彩認証システムのプロトタイプ開発を行った。

図 組込み向け虹彩認証システム開発ボード



虹彩認証システムで使用した関数ライブラリは、以下の通り。

ライブラリ名	ライブラリ識別名
画像サイズ変更	<i>ipGtResizeImgBilinearIf</i>
ヒストグラム均一化処理	<i>ipFilEqualizeHistIf</i>
局所的ヒストグラム均一化処理	<i>ipFilLocalEqualizeHistIf</i>
グレースケール変換	<i>ipCcCvtGrayIf</i>
BLPOC	<i>ipCorBLPOCIf</i>
低周波成分切出し	<i>ipCorCutHighFreqIf</i>
正弦波テーブル生成	<i>ipSinGenSinIf</i>

画像データの行列転置	<i>ipArrTransposeIf</i>
バイリニア補間	<i>ipIplBilinearIf</i>

3. 3D 顔復元／リアルタイム 3D 顔復元システム

標準ライブラリや GPU を対象として開発した GPU 版ライブラリの妥当性検証のため、3D 顔復元システム、リアルタイム 3D 顔復元システムのプロトタイプシステムの開発を行った。

図 3D 顔復元システム ワイヤーフレーム表示

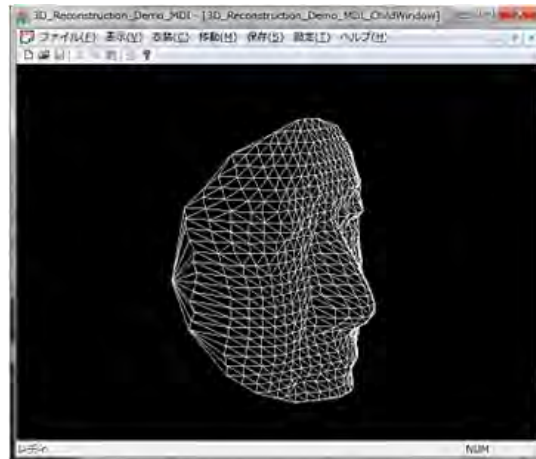
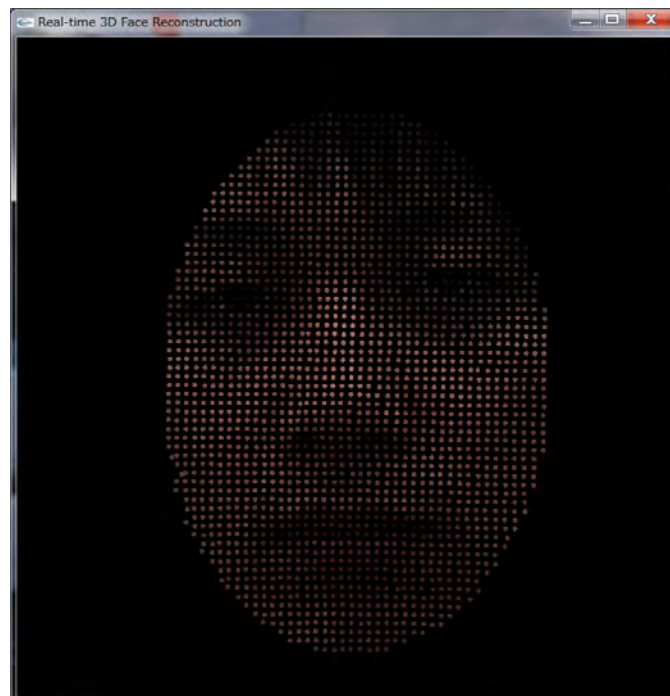


図 リアルタイム 3D 復元システム



3D 顔復元／リアルタイム 3D 顔復元システムで使した関数ライブラリは、以下の通り。

ライブラリ名	ライブラリ識別名
Matlab パラメータによる簡易画像平行化変換	<i>ipGtParallelSimpleIf</i>
グレースケール変換	<i>ipCcCvtGrayIf</i>
二次元階層ブロックマッチング	<i>ipMchDetectHBM2dIf</i>
高速 1 次元階層ブロックマッチング	<i>ipMchDetectHBM1dAceIf</i>
顔検出	<i>ipOdFaceIf</i>
三角測量	<i>ipScDistTriangulationIf</i>
三次元座標算出(平行化済み画像用)	<i>ipScReconstruct1dIf</i>
三次元座標算出(非平行化画像用)	<i>ipScReconstruct2dIf</i>
三角形リスト作成	<i>ipScMakeTriangleIf</i>

4. 3D ピッキングシステム

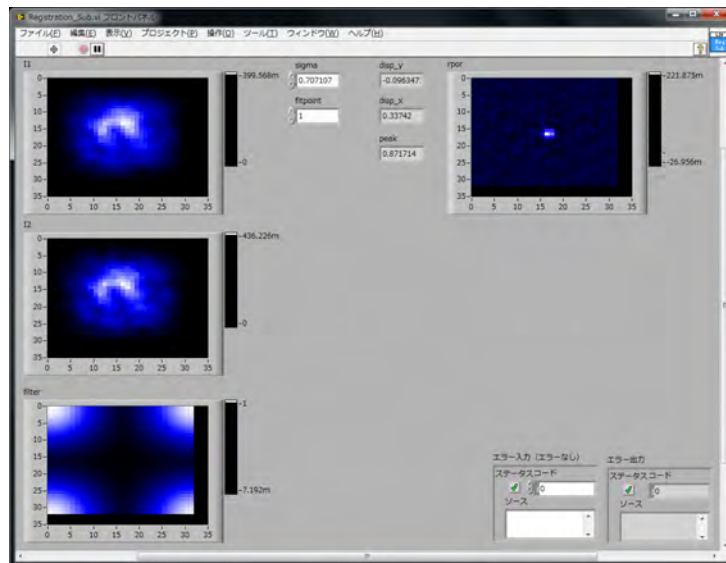
POC、*HBM* のアルゴリズムを、*FA* 分野のシステムへ流用する際の実用性を検証するために、物体の一致判定と 3 次元計測を行うピッキングシステムの開発を行った。

(1) *POC* を用いた物体の一致判定

POC から得られる類似度を用いることで、撮影した対象物と予め登録されている対象物の一致判定が可能であることを確認した。

また、2 枚の画像に回転角度が発生する場合、そのままの画像では *POC* による一致判定が行えないため、画像の回転角度を補正する前処理を行うようにしている。

図 POC 関数画面



(2) *HBM* を用いた物体の 3 次元計測

2 台のカメラで撮影した画像から、*HBM* で得られる対応点探索結果を元に、対象物との距離計測が可能であることを確認した。

また、計測精度を確認するため、対象物距離が $280(\text{mm})$ 、 $380(\text{mm})$ の距離でそれぞれ距離計測を連続 10 回行い、実測距離との誤差を算出し、誤差の最大値、最小値、平均値を求めた。(表)

集計した誤差の値から、*HBM* を用いた距離計測の精度は 1mm 以下であり、今回のピッキングシステムにおいて、仕様を十分満たせることがわかった。

図 計測精度検証構成図

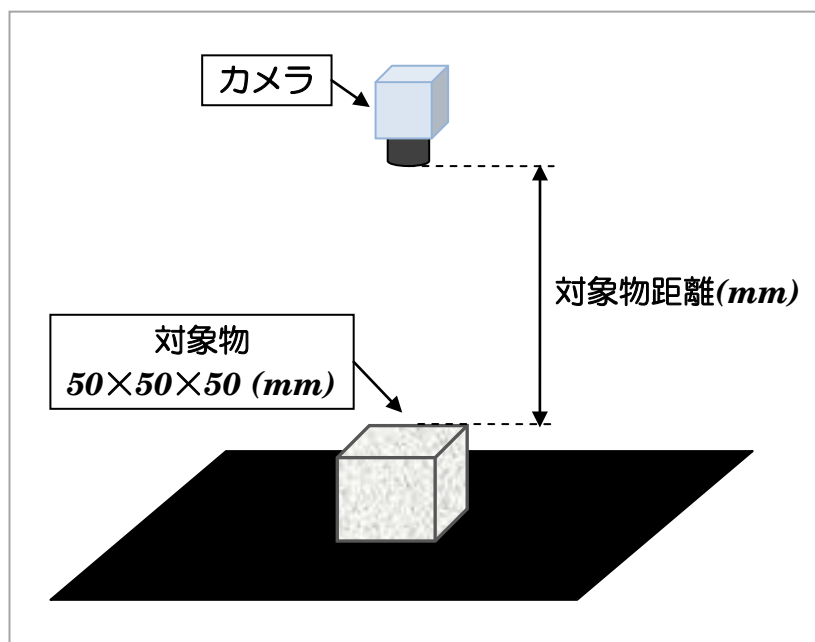
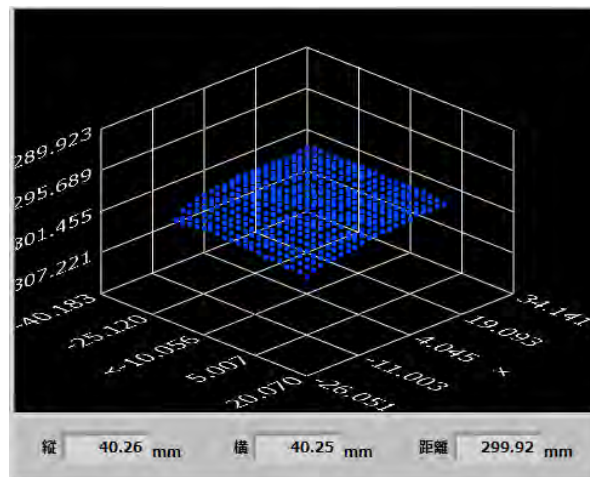


表 実測距離との誤差

対象物距離 (mm)	誤差(mm)		
	最大値	最小値	平均値
280	0.98	0.69	0.91
380	0.89	0.85	0.87

図 3次元計測結果画面



(3) (1)～(2)の機能を実装したピッキングシステムの開発

(1)、(2)のアルゴリズムを用いて、乱雑に置かれた複数の物体から、予め登録された物体をピッキングし、指定されたエリアに物体を整列させるピッキングシステムの開発を行った。

ピッキング動作にはロボットを使用し、3次元計測で行った物体との距離を用いることで、ロボットハンドで物体を取得・設置する動作を実現している。

また、ピッキングシステムでのロボット制御用に、ロボット制御 *BOX* を作成した。

図 ピッキングシステム（メイン画面）

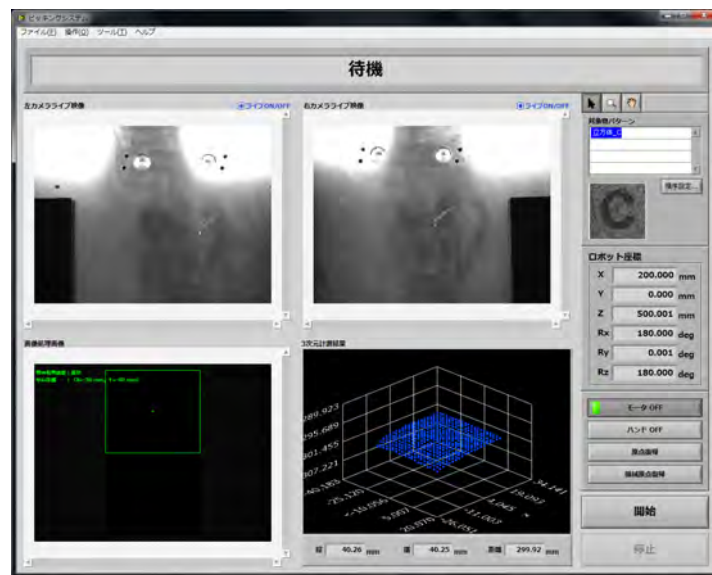


図 ピッキングシステム全体図

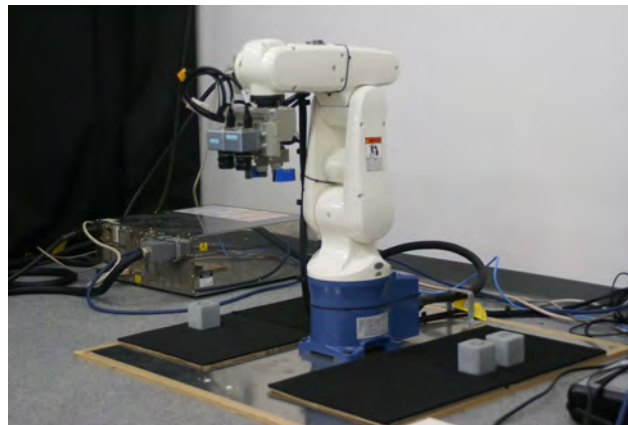


図 ロボット制御BOX



④-2プロセスの妥当性評価

プロトタイプ開発で得られた各種指標を元に本開発技術手法の妥当性評価を行う。

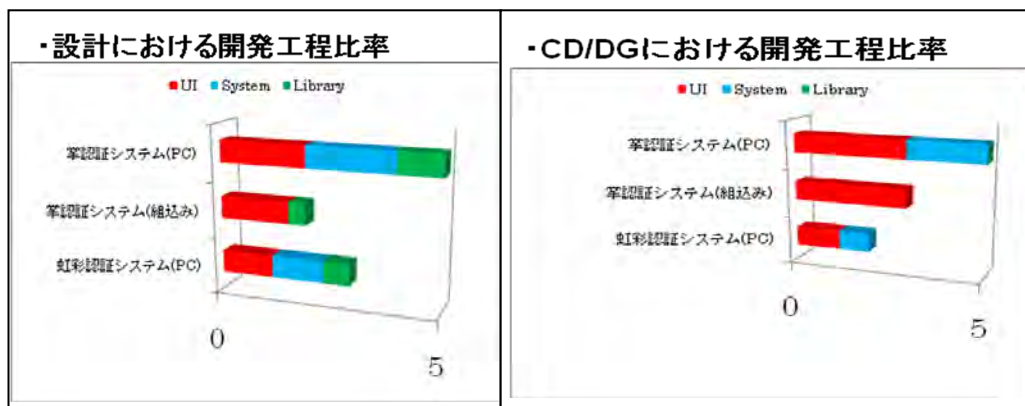
1. 開発効率

プロトタイプとして作成した以下の3つのシステムの、実装における開発工数と開発工程比率を下表に示す。

表. 開発工数

	掌認証システム (PC)	掌 認 証 シ ス テ ム (組 込 み)	虹彩認証システム (PC)
設計	5人日	2人日	3人日
CD/DG	5人日	3人日	2人日

表. 開発工程比率



UI : 外部 UI とのインタフェース部分
 System : モデルコアのカスタマイズを行う部分
 Library : 共通インタフェース呼び出し部分

プロトタイプ開発は、掌認証システム (PC) → 掌認証システム (組込み) → 虹彩認証システム (PC) の順で行われた。

掌認証システム (PC) の開発には認証モデルコアをベースとした開発が必要であったため、全体として工数もかかっており、工程比率も UI の開発だけではなく、System の開発が発生している。次に、掌認証システム (PC) と掌認証システム (組込み) を比較すると、System は等しいため開発は行っていない。その代わりに、組込みプラットフォームに特化した UI の開発が必要であるため、UI の開発に注力している。最後に、掌認証システム (PC) と虹彩認証システム (PC) を比較すると、開発工程の比率は等しくなっているが、開発工数が半分となっている。これは、掌認証システム (PC) において、認証モデルを利用した開発のノウハウの蓄積により、そのノウハウを利用したことによって開発効率が向上したためである。

全体を通して、Library の開発は利用する上での調査が主であり、ライブラリ本体の開発は行っていない。これは MTOP におけるライブラリ

資産の有効性が証明されたといえる。

したがって、モデルコアとプラットフォーム固有のライブラリを組み合わせることによって目的のシステムの構築が短期間で可能となり、開発においてはシステム固有の部分と *GUI* の部分の設計および開発に注力することが可能となる。

これにより、プラットフォームに依存しないモデルコアと、プラットフォームごとに用意されたライブラリの組み合わせによる *MTOP* 開発の有効性を証明することができたといえる。

2. 不具合率

虹彩認証システム開発を通してのライブラリの不具合件数は 10 件、使用ライブラリのステップ数から不具合発生率は、0.5%（10 件／約 2000 ステップ）と低く、当初目標としていた指標を充分満たす結果となった。

また、3D 顔復元／リアルタイム 3D 顔復元システム開発を通してのライブラリの不具合件数は 14 件、使用ライブラリのステップ数から不具合発生率は、0.4%（14 件／約 3800 ステップ）程度と低く、こちらも虹彩認証システム同様、当初目標としていた指標を充分満たす結果が得られた。

3. 精度

POC、*HBM* は、特徴が無い対象物に対して計測を行うことが難しいため、表面が平坦、または鏡面になっている対象物は、計測の対象外となってしまう。よって、一致判定や 3 次元計測を行う対象物には、特徴点の抽出が可能であるということが条件となるため、*FA* 分野では特徴が無い対象物を取り扱うことが多いことから、現状はシステムとして導入することは難しいと考えられる。

また、外乱（外光）によって計測精度にバラつきが発生したため、照明を用いることにより、安定したシステム環境の構築が必要である。

今後の課題としては、特徴が無い対象物に対する一致判定や 3 次元計測を実現させるために、どのようなアルゴリズムが必要かを、引き続き東北大と共同で検討・調査を行う必要がある。この課題を解決することにより、アルゴリズムの汎用性が向上し、*FA* 分野のシステムへの実用性も向上することが考えられる。

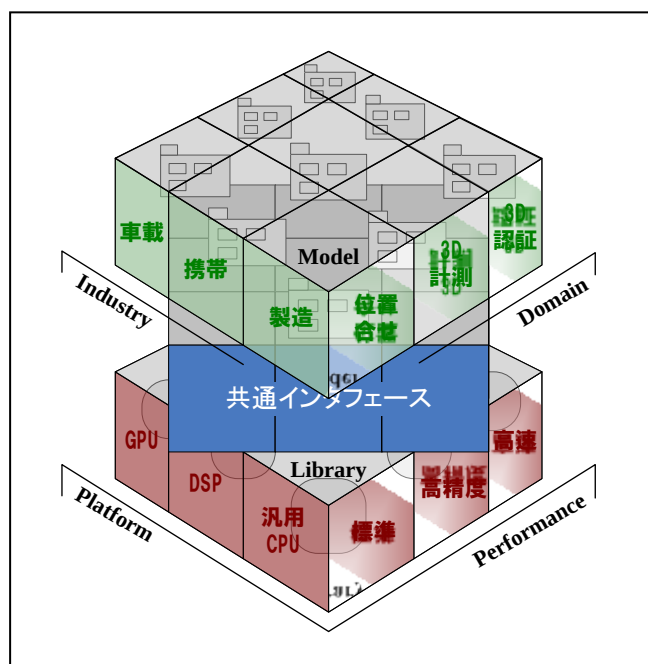
第3章 全体総括

1. 研究開発成果

本研究開発は、大学の研究資産を産業分野へ早期にシーズアウトすることを目的として、「モデル駆動開発」「ソフトウェア資産化」「研究アルゴリズムの実装」「高性能ハードウェアに対応したライブラリ開発」など、これまで実現できていない手法を、新たに提起した開発技術手法をもとに一貫したプロセスとし、実際のプロトタイプ開発でその妥当性を検証するものである。今回研究資産からソフトウェアへと資産化したボリュームは商用ライブラリに比べれば決して多くはないが、東北大が有する主要な要素技術と基本機能を含み、汎用 PC だけでなく高性能ハードウェアを搭載した各種特定用途向けへと適用の幅を広められたことは、今後広範囲に事業展開する上で重要である。プロトタイプ開発においては、モデル駆動開発を使って、従来からの部品であるライブラリに加え今後主流になるモデルを新たな部品として融合することにより、使いやすく抽象度の高いソフトウェア資産としての有効性を証明した。本再利用手法は、*ESEC2010* における資料展示でツールメーカーやレガシー資産を多数保有する機器メーカー、ソフトベンダーから評価を得ている。

本研究開発で開発されたソフトウェア資産はマシンビジョン技術を必要とする様々な市場に柔軟に適用できるように構成されている。具体的には、対象とする産業のマーケットに合わせてモデルコア（アプリケーション）を選択し、技術的要件である性能・ハードウェア要件に合わせてライブラリを選択し、あたかもルービックキューブのように組み合わせることが可能である。

図. ソフトウェア資産概念図



本研究開発では、以下に示すような規約やガイドライン、マニュアルなどのドキュメント類からソフトウェアまでの有形の成果を配し、そこに至るまでの多くの無形の成果を蓄積できた。本成果は本開発技術手法とソフトウェア資産を中心として永続的に繰り返し利用され価値向上していくものと期待する。

表. 研究開発成果物一覧

研究項目	ドキュメント	ソフト・システム	その他
①開発技術手法	基本設計書 MTOP アーキテクチャ定義書	—	—
②ライブラリ開発	ライブラリ開発ガイドライン コーディング規約 開発 Tips 共通関数説明書 ライブラリ一覧 スペックシート プラットフォーム適合表	汎用 CPU 版ライブラリ 組込み向ライブラリ GPU 版ライブラリ DSP 版ライブラリ (OpenCV 版ライブラリ)	—
③モデルコア開発	モデルコア開発ガイドライン モデリング規約 モデルコア開発 Tips リファレンスマニュアル モデルコア一覧 スペックシート プラットフォーム適合表 チュートリアル	認証モデルコア 掌認証モデルコア 虹彩認証モデルコア	サンプルモデル
④プロトタイプ開発	各種仕様書 各種説明書	掌認証システム 虹彩認証システム 顔復元システム リアルタイム顔復元システム ビッキングシステム	—

2. 今後の課題及び事業化展開

(1) 今後の課題

本研究開発における今後の課題は、以下の4点である。

- ① 性能向上、バリエーションの拡充
- ② モデリングツール利用環境
- ③ ソフト資産の利用促進と保守

今回開発したライブラリは、汎用 PC 版のベンチマーク測定による性能比較では性能差が見られた。当初の計画から汎用 PC 版は実現可能性を検証するうえでの利用用途の広いライブラリと位置付けており、特定用途向けライブラリをもって産業分野のオーダーに合わせる予定であったが、プロトタイプ開発での評価では、分野により達成できていない状況が見受けられた。また、バリエーションについても時間的制約からプロトタイプに合わせたライブラリ開発となった経緯がある。非特定の分野で早期に試作化が可能となるよう、今後も広範囲のプロトタイプ開発を経てフィードバックをかけながらソフトウェア資産の蓄積を進める必要がある。

今後の事業展開を見たうえで、②のモデリングツールの利用環境は深刻である。今回の研究開発でもモデル駆動開発と画像処理ライブラリを組み合わせた開発効果は高く評価することができ、地方の組込みソフトウェア産業を見たとて地場企業が全国的に見ても先進的なモデル駆動開発を武器に事業展開することは地域色を出すうえでも非常に好ましい。しかし、これらツールは商用利用する際には非常に高価であり、中小企業が利用するには負荷が高く、このことが日本のソフトウェア産業の遅れの一要因になっていると言っても過言ではない。この環境制約を地域の産官学の連携をもって共同で環境を創ることにより、中小企業が新しい技術利用に取り組める仕組みづくりが必要である。

最後に、今回開発したソフトウェア資産は①にもあるように熟成の必要がある。その為にはソフトウェア資産を多くの企業によって開発に利用してもらい、フィードバックシステムを活性化させる必要がある。その為に柔軟なライセンス設定と開発パートナー向け環境の提供を進める。また、将来的にはソフトウェア資産の保守体制も必要であるが、当面は開発パートナー企業が協力体制を組んで対応にあたりソフトウェア資産の充実を目指す。

(2) 事業化計画

近年の傾向として、工業や医療などの分野においてこれまで人間の目で行ってきた官能的な作業を画像処理を用いたマシンビジョンの技術を用いる傾向が強くなってきている。また、家電やモバイル、エンターテインメントの分野

では3D技術の取り込みがはじまり、今後さらに市場は拡大すると予想される。これらのことから多様な応用を求める分野からのマシンビジョン技術のニーズが増えると予想され、研究から試作、製品開発の上で本ソフトウェア資産を活用した迅速な開発が強みとなると考えている。

<事業展開の構想>



①研究開発へのアプローチ

H22 年度戦略的高度化支援事業においてF A分野に特化した研究開発を新たに進める。課題として残るライブラリの性能をさらに向上し、マシンビジョン技術の導入がニーズとして強く存在する外観検査に対し最適なアルゴリズムの研究と官能検査を可能とするソフトウェアの開発を進める。

②試作開発へのアプローチ

モバイル分野へ高度なバイオメトリクス認証技術を適用する研究開発を進めている（詳細はNDAにより非公開）。試作開発において本ソフトウェア資産を活用し検証サイクルを向上させ効率的な研究開発を進める。

③デモンストレーションへのアプローチ

デジタルサイネージ市場に対し小規模店舗向けインタラクティブ型のシステム開発を地元企業で検討している。本ソフトウェア資産を用いて画像処理システムを組み込むことにより低コストで柔軟なシステムの構築が可能である。製品販売を視野に入れたプロトタイプ開発を本年度中に計画している。

付録

1. 専門用語の解説

専門用語	解 説
MATLAB：マトラボまたはマトラブ	MathWorks 社が開発した数値解析ソフトウェアであり、その中で使うプログラミング言語の名称。手軽に数値解析を行えることから、シミュレーションを行う際の数値解析ソフトとして研究機関や、製品プロトタイプ試作時のシミュレーション等に使用されている。MATLAB は「信号処理・通信」「データ解析・アルゴリズム開発」「画像処理」「制御系設計」「実験計測」「コンピュータを利用した生命工学」「金融モデリングおよび解析」といった様々な分野での数値計算に対応している。そのため、世界中の大学及び研究機関、また国内においても 300 以上の研究機関で数値科学計算のための標準ツールとして導入されており、デファクトスタンダードとなっている。
M-file	MATLAB の実行文を含んだ拡張子.m を持つテキストファイル。MATLAB で使用する関数や、スクリプトを記述することにより、作成した関数、スクリプトの再利用が可能である。
MEX-file	MATLAB のスクリプトやコマンドラインから呼び出せる C 言語や Fortran のバイナリファイル。MEX-file 化することにより、C 言語や Fortran で書かれたプログラムの流用や、計算コストの高い部分を C 言語で書くことによる計算の高速化や消費メモリの節約が可能である。
LabVIEW	NI（ナショナルインスツルメント）社製の開発言語。G 言語と呼ばれるグラフィカルなプログラミング言語で、計測・制御システム構築に秀でている
UML	UML（統一モデリング言語、Unified Modeling Language）は、ソフトウェア工学におけるオブジェクトモデリングのために標準化した仕様記述言語である。UML は、グラフィカルな記述で抽象化したシステムのモデル（UML モデル）を生成する汎用モデリング言語である
OpenCV	OpenCV とはインテルが開発・公開したオープンソースのコンピュータビジョン向けライブラリである
Rhapsody: ラプソディー	IBM (旧 Telelogic) 社が開発したモデル駆動型開発 (MDD) 環境。UML 2.1、SysML、DoDAF、AUTOSAR を使って、システム設計とソフトウェア開発を行うことが可能で、モデルから C、C++、Java、Ada のアプリケーションコードを自動で生成可能である。
モデル駆動型開発 (MDD : Model Driven Development)	UML などを用い対象領域をモデル化したものを変換することによって、最終的に CPU 上で実行可能なモデルとするソフトウェア開発手法のこと。MBE:Model Based Engineering, MDA:Model Driven Architecture と呼ばれる。
POC	位相限定相関法 (POC) はフーリエ変換された信号の位相成分のみに着目する画像マッチング手法。振幅情報を用いた従来の 2 次元相関法や特徴抽出法とは全く異なり、外乱に強く大きな誤りがないという特徴を持っている
HBM	東北大学青木教授が提案するステレオ画像のサブピクセル対応付け手法である。位相情報を元に画像マッチングを行うため「ロバスト性」と「位置合わせ精度」の観点から特に優れた特性を示す

2. ライブラリー一覧

カテゴリ	関数名	関数略称	説明
特徴点検出 FeatureExtract	SURF 検出	ipFeSURFIf	SURF を用いた特徴点と、その特徴点における特徴ベクトルの検出を行う
	U-SURF 検出	ipFeUprightSURFIf	回転方向の変化を考慮しない SURF 検出を行う
サンプリング Sampling	相似線形領域取得	ipSamGetSimilarArealf	元画像の指定された座標を中心とした、回転角を考慮した指定サイズの領域を切り出す
	画像サイズ変更	ipGtResizeImgBilinearlf	バイリニア補間による画像のサイズ変更を行う
	画像アフィン変換	ipGtCvtAffinelf	画像のアフィン変換を行う
画像変換 (幾何変換) Geometrical Transforms	座標変換	ipGtHomogeouself	座標を同次座標(または同次座標から座標)に変換する
	ホモグラフィ行列 (平面射影変換行列)	ipGtHomographyRansacIf	2 枚の画像間の平面射影行列を、RANSAC アルゴリズムにより求める
	OpenCV パラメータによる画像平行化変換	ipGtParallelCvlf	OpenCV で算出されたパラメータを用い、画像に対してステレオカメラの歪み補正及び平行化変換する
	Matlab パラメータによる画像平行化変換	ipGtParallelMatlf	Matlab を使用したカメラキャリブレーションより算出されたパラメータを用い、画像に対してステレオカメラの歪み補正及び平行化変換する
	Matlab パラメータによる簡易画像平行化変換	ipGtParallelSimplelf	Matlab を使用したカメラキャリブレーションより算出されたパラメータを用い、画像に対してステレオカメラの簡易的な画像平行化変換を行う
	平滑化	ipFilSmoothlf	指定された方法で画像の平滑化を行う
フィルタ Filter	ヒストグラム均一化処理	ipFilEqualizeHistlf	画像のヒストグラムを均一化する
	局所的ヒストグラム均一化処理	ipFilLocalEqualizeHistlf	画像のヒストグラムを局所的に均一化する
	ヒストグラム計算	ipFilHistlf	画像のヒストグラムの計算を行う
	閾値処理 (2 値化処理)	ipFilThresholdlf	シングルチャンネルの2値化を行う
	閾値処理 (範囲指定版)	ipFilThresholdByRangelf	シングルチャンネルの入力配列に対して、指定された範囲を閾値とした 2 値化処理を行う
	メディアンフィルタ (1 次元配列版)	ipFilMedianFilter1Dlf	指定されたフィルタ長にて、入力データにメディアンフィルタ処理を行う
	画像の収縮処理 (Erosion)	ipFilErosionlf	画像の収縮処理を行う
	画像の膨張処理 (Dilation)	ipFilDilationlf	画像の膨張処理を行う
	オープニング	ipFilOpeninglf	収縮→膨張を行い、孤立点を除去する
	クローズング	ipFilClosinglf	膨張→収縮を行い、離散点を結合する
	Bayer 配列の変換	ipCcBayerToGraylf	入力画像(Bayer 配列)をグレースケール画像に変換する
	グレースケール変換	ipCcCvtGraylf	画像のグレースケール変換を行う
色変換 Color Conversion	カラー変換 (RGB)	ipCcCvtRGBlf	BMP(RAW)から RGB へ変換する
	カラー変換 (HSV)	ipCcCvtHSVlf	RGB から HSV へ変換する
	輪郭検出	ipCdGetContourslf	2 値画像の輪郭の個数を調べる
	チェーンコード	ipCdCreateChainCodePalmlf	チェーンコードの生成を行う
輪郭検出 ContoursDetection	1 次元階層ブロックマッチング (HBM)	ipMchDetectHBM1dlf	1 次元による画像特徴点の階層ブロックマッチングを行う
マッチング Matching	2 次元階層ブロックマッチング (HBM)	ipMchDetectHBM2dlf	2 次元による画像特徴点の階層ブロックマッチングを行う
	高速 1 次元階層ブロックマッチング (HBM)	ipMchDetectHBM1dAcelf	1 次元による画像特徴点の単精度階層ブロックマッチングを行う
	オブジェクトを検出	ipOdObjectUseSURFIf	画像中のオブジェクトを SURF アルゴ

	(SURF)		リズムを用いて検出する
	顔検出	ipOdFacelf	画像中の顔を検出する
キャリブレーション Calibration	エピポーラ線の計算	ipScEpiLinf	ステレオ画像上のエピポーラ線を計算する
	三角測量	ipScTriangulationlf	視差画像から各点の距離を算出する
ステレオマッチング StereoCorrespondence	3次元座標算出 (1次元用)	ipScReconstruct1dlf	平行化されている画像の座標から、3次元空間点の座標を計算する
	3次元座標算出 (2次元用)	ipScReconstruct2dlf	2枚の2次元画像の対応点群の座標と各画像のカメラへの射影変換行列から3次元空間点の座標を計算する
	三角形リスト作成	ipScMakeTrianglelf	立体表示するためのメッシュ形成用の三角形リストを作成する
	FFT /IFFT 領域の確保	ipSpFFTMallocf	FFT/IFFT をかける入出力配列のメモリ確保を行う
	FFT/IFFT 設定 (1次元)	ipSpFFTPlan1dlf	1次元 FFT/IFFT の設定を行う
信号処理 SignalProcessing	FFT/IFFT 設定 (2次元)	ipSpFFTPlan2dlf	2次元 FFT/IFFT の設定を行う
	FFT/IFFT の実行	ipSpFFTExecutelf	FFT/IFFT 設定に基づき、FFT/IFFT を実行する
	FFT/IFFT 設定の解放	ipSpFFTPlanDestroylf	FFT/IFFT 設定の領域を解放する(1次元/2次元共通)
	FFT /IFFT 領域の解放	ipSpFFTFreef	FFT/IFFT をかける入出力配列のメモリ確保を行う
	回転因子計算処理	ipSpCalTwiddleTablelf	回転因子を計算する
	FFT Shift	ipSpFFTShiftlf	直流成分を中央に移動する処理を行う
	IFFT Shift	ipSpIFFTShiftlf	中央に移動された成分をもとに戻す処理を行う
	FFT/IFFT 演算	ipSpCalculateFFTIlf	2次元離散/逆離散フーリエ演算を行う
	位相データ取得	ipSpGetFrequencyDataIf	位相データの取得を行う
	クロスパワースペクトル算出	ipSpCrossPowerSpectrumlf	クロスパワースペクトルの算出を行う
	1次元 POC	ipCorPOC1dlf	1次元 POC による平行移動量の高精度検出を行う
	2次元 POC	ipCorPOC2dlf	2次元 POC による平行移動量の高精度検出を行う
相関関数 Correlation function	2次元簡易 POC	ipCorPocComplf	FFT 処理済みデータに対する POC を行う
	帯域制限位相限定相関法 (BLPOC)	ipCorBlpoclf	類似度の評価指標 (帯域制限 POC) を作成する
	低周波成分切出し	ipCorCutHighFreqIf	FFT を行い指定された低周波成分を切り出す
	最大値検出	ipCorFindPeakValueIf	入力複素数データから最大値 (ピーク値) を検出する
	正弦波テーブル生成	ipSinGenSinlf	正弦波テーブルを作成する
	画像データの行列転置	ipArrTransposeIf	画像データに対して行列転置を行う
正弦波 Sine wave	バイリニア補間	ipPlBilinearlf	バイリニア補間を計算する
配列操作 Array Transpose	グレースケール画像データ構造体の領域確保	ipIniCreatelmagelf	画像構造体の領域を確保する
補間 Interpolation	グレースケール画像データ構造体の領域解放	ipIniReleaself	画像構造体の領域を解放する
初期化 Initialization	カラー画像データ構造体の領域確保	ipIniCreateCollmgIf	カラー画像構造体の領域を確保する
	カラー画像データ構造体の領域解放	ipIniReleaseCollmgIf	カラー画像構造体の領域を解放する
	多重行列構造体領域確保	ipIniCreateMatrixlf	多重行列構造体の領域を確保する
	多重行列構造体領域解放	ipIniReleaseMatrixlf	多重行列構造体の領域を解放する

3. モデルコア一覧

モデルコア名	ターゲット	パッケージ名	説明
認証モデル	ベースモデル	UI	認証システム全体を制御
		RecognitionSystem	認証登録・照合状態を管理
		ExtensionRecognitionSystem	認証のための画像処理、実際の認証処理を実行 〈カスタマイズ必須箇所〉
		ImageLibrary	画像処理ライブラリへのアクセス 〈カスタマイズ可能箇所〉
		Common	画像データクラス定義
	掌認証	UI	ベースモデルと同じ
		RecognitionSystem	ベースモデルと同じ
		ExtensionRecognitionSystem	機能拡張～掌用～ 掌認証用カスタマイズ
		ImageLibrary	ベースモデルと同じ
		Common	ベースモデルと同じ
	虹彩認証	UI	ベースモデルと同じ
		RecognitionSystem	ベースモデルと同じ
		ExtensionRecognitionSystem	機能拡張～虹彩用～ 虹彩認証用にカスタマイズ
		ImageLibrary	ベースモデルと同じ
		Common	ベースモデルと同じ