位置情報付き写真データを用いた携帯電話向け移動支援システムとその実現法

任貴全† 浦田繁玄† 梅津高朗† 中田 明夫† 安本 慶一‡ 東野 輝夫†

† 大阪大学大学院情報科学研究科

‡ 奈良先端科学技術大学院大学情報科学研究科

GPSから取得した位置情報を付加した写真をサーバを介して共有することで，ナビゲーションを行うパーソナルナビゲーションシステムとその効率的な実装法を提案する．ユーザは写真を撮影し，位置情報と共にナビゲーションサーバに登録する．サーバはそれらの写真を地図と共に他のユーザに配布するサービスを提供する．携帯端末はメモリなどの制限から実行できるプログラムのサイズなどに制限がある．そこで，最も効率的に実行が行えるようにサーバ，クライアントにプログラムを分割する手法を提案する．分割した場合の通信量や計算量をシミュレーションにより見積もり，それらを最適化するようサーバ，クライアントの機能を決定することで効率的な実装を行える手法を提案した．

A Personal Navigation System Using Photos with Location and Its Efficient Implementation in Cellular Phones

Ren GuiQuan†, Urata Shigeharu†, Umedu Takaaki†, Keiichi Yasumoto‡, Akio Nakata† and Teruo Higashino†

† Graduate School of Info. Sci. & Tech., Osaka Univ., Japan

‡ Graduate School of Info. Sci., Nara Institute of Sci. & Tech., Japan

In this paper, we present design and implementation of a personal navigation system, which shows users the photos corresponding location provided from GPS. Users can register photos needed for navigation with their location to the navigation server. For the other users, the server provides the map data around them and the photos taken in the area. The mobile terminals cannot execute large programs because of their small resources. Then we propose a method to divide the system such that server and clients can cooperate most efficiently. We have estimated the performance of divided systems such as the amount of communication by simulation, and optimized the division.

1 はじめに

近年，携帯電話やPDAのような携帯端末が急速に普及してきており，また様々な機能が付加されるようになっている．特に，GPS衛星を利用した位置情報の取得機能は，ネットワーク上の端末と実空間における座標とを結びつけることができるため，従来とは異なる新しい分野のアプリケーションに応用できるとして注目されている．その一つとしてカーナビゲーションシステムから発展したパーソナルナビゲーションシステムに関する研究が盛んに行われている．しかし，現在最もよく使われている携帯電話では，端末の能力の制限から，カーナビゲーションシステムようにデータ量の大きい地図情報をそのままの形で携帯電話に搭載して利用することができない．そこで，本研究では，本来，携帯端末で実行不可能なサイズのアプリケーションの一部をサーバで実行し，携帯端末と協調動作させることで大きなサイズのアプリケーションを携帯電話端末上で実行させる手法を用いてパーソナルナビゲーションシステムを設計，実装する．

JavaにはJava RMI [1] やHORB [2] などといった遠隔呼び出しの方法が用意，提案されており，こういった遠隔メソッド呼び出しを用いることで複数端末により機能や負荷の分散化を行う様々な方法が研究されている．本研究ではパーソナルナビゲーションシステムを携帯端末において遠隔メソッド呼び出しを用いて実装する．

提案するパーソナルナビゲーションシステムでは，ユーザが撮影した写真を用いてナビゲーションを作成する．まず，作成するユーザが目的地までの経由地を訪れ順に写真を撮影し，ナビゲーションシナリオを作成する．その際に，GPSを用いて同時に位置情報が取得され，写真と共にサーバに転

送され，登録される．ナビゲーションを受けたい別のユーザは，自分の位置を取得して，サーバに問い合わせることで，ナビゲーションシナリオをロードする．その際，携帯電話に対してシナリオ全体のデータをロードすることはできないため，使用頻度の高い地図や案内用写真のみを事前にロードしておき，移動に伴いインタラクティブに新たなデータを取得していく．

Javaによる遠隔メソッド呼び出しに関しては，性能解析と最適化に関する研究 [3] や，実時間システムの性能向上ための性能計測やスケジューリングに関する研究 [4, 5]，また，RMI をより効率的に実装するための研究 [6] など様々な研究がなされている．ここでは，携帯電話の環境を考慮し，より効率的な実装を行うため，環境毎に自動的に最適な機能分割，配置が得られるような手法を提案する．

機能の配置を考える場合，分割の際の基準となる通信量や通信回数などの統計情報が必要となる．こういった統計的な評価としては，並行 Java プログラムのスライシングに関する研究 [7, 8] 等の様に呼び出し関係をプログラムのソースコードを解析して行う手法も存在するが，本研究においてはシミュレーションに基づく性能計測を用いる．統計情報を収集できるよう修正を加えたプログラムを様々な状況を考慮して各々一定時間動作させることで，それらの情報を収集する．

次に得られた統計情報に基づき，携帯端末のメモリ容量を超えない範囲で要求された性能が最良となるような分割を求める．この分割問題は NP 困難問題であり，現実的な時間で最適解を求めることは困難であるため，SA(Simulated Annealing) 法 [9] を用いて近似解を求めることとした．

2 提案するパーソナルナビゲーションシステム

2.1 ナビゲーション作成機能

ナビゲーション作成手順は以下の通りとなる．

（1） 出発地，目的地およびナビゲーション利用者を導く経路を決定
（2） 経路上でで重要となる地点に行き携帯電話で写真を撮影
（3） 写真に対して GPS 情報を付与
（4） 写真に対してコメントを付与
（5） 写真をサーバに登録
（6） 必要な地点の写真を全て登録しナビゲーション作成完了

ナビゲーションを作成する場合，出発地と目的地の間をどのような経路で誘導するかを決定する．そして実際にその場所に行き，経路上でナビゲーション利用者が迷いそうなポイント (駅での乗り換えや出口の方角など) の写真を撮影する．提案システムでは，撮影された写真には自動的に GPS から取得した位置情報が付加される．それらの写真を簡単なコメントと共にサーバに登録する．サーバは，対象とする範囲の地図情報を保持しており，登録された写真の位置情報に基づき，その写真を記録する．登録された写真は登録順に地図上に配置されナビゲーションシナリオとして記録される．

2.2 ナビゲーション参照機能

ナビゲーションを参照するプログラムは以下のように動作する．

（1） 出発地，目的地を選択
（2） GPS により位置情報を取得
（3） 周辺の地図および近隣の登録ポイントの写真，コメントなどを取得
（4） 携帯電話の移動に従って経路上のデータを順次取得

ナビゲーションを受けたいユーザはサーバに自らの位置情報を送信し，他のユーザの作成したナビゲーションシナリオに従いデータを取得する．地図上に配置された写真とコメントを順に参照することで，目的地までの道のりの把握に利用できる．

携帯端末ではメモリの制限からシナリオ全体を一度に取得することはできないため，インタラクティブに順次取得する必要がある．また，地図の画像データは一般にベクトルデータとして得られる．ベクトルデータの形で転送した場合には携帯端末に生じる CPU 負荷が，レンダリング後に転送した場合には通信コストがそれぞれ増大するため，その間のトレードオフを考慮する必要がある．携帯電話は機種や世代の違いにより利用できるメモリ量や CPU パワー，通信コストなどが異なり，機種毎にそれぞれ最適なプログラムを用意することは多くのコストを必要とする．そこで，ここでは，システム全体を単体のアプリケーションとして記述し，その一部をサーバに配置した場合にどのような性能となるかの見積もりを行い，目的関数を与えることで自動的に最適なサーバクライアントの機能割り当てを行う手法を適用した．

3 システムの実装法

3.1 分割の最適化のための統計情報見積もり手法

まずシステムをサーバ側と携帯電話側のモジュールに分割する際の基準となるデータを求めるため，与えられたアプリケーションに必要となる統計情報を記録するコードを挿入し一定時間動作させることでシミュレーションを行う．状況に応じて最適な分割が行えるよう，複数の項目について見積もりを行い組み合わせて利用する．ここでは，分割の基準とする項目として，**サーバ・端末間の通信量，通信遅延，各モジュールで消費される CPU 時間**を考える．携帯端末は一般的にサーバに比べて CPU のリソースが劣るため，携帯端末で消費される CPU 時間が小さくなるよう分割を行うことで全体の実行速度を高速にでき，また，携帯端末で消費される電力量を低減することができる．ただし，一般的には生じる通信量等は対象アプリケーションの毎回の実行によって変化する．分割を行う際の入力として用いる場合は，実際の使用状況に近い状態で複数回実験を行い平均的な値を求める必要がある．

以下，各統計量の見積もり方法について述べる．

3.1.1 通信量の見積もり

分割した際の通信量を見積もるため，全てのメソッド呼び出しが遠隔呼び出しされたと仮定した場合のデータを収集する．

3.1.2 通信遅延の見積もり

ここでは遅延時間は主に通信オーバーヘッドにより発生すると仮定し，遠隔メソッド呼び出しの回数を用いて評価する．実際には通信量や回線速度に応じて変化するため，実環境に応じて通信量の計数との重み付けを調整する必要がある．

3.1.3 CPU 時間の見積もり

CPU 時間は各クラスに関してそのクラスに属するメソッドの総実行時間を計測することで見積もりを行う．各メソッド呼び出し時にその時刻を記録し，呼び出しが終了した (もしくは，そのメソッド内から標準 API 以外の他のクラスのメソッドが呼び出された場合にはその呼び出しが発生した) 時刻との差分を累計することでそのメソッドの属するクラスの CPU 時間とする単純な手法を用いた．

3.2 分割問題の定式化

ここでは，以下のように分割問題の定式化し，最適化を行った．

- モジュール数 $n \in \mathbf{N}$
- モジュールの集合 $M = \{m_1, ..., m_n\}$
- 各モジュールの使用メモリサイズ $S : M \mapsto \mathbf{N}$
- 端末の最大メモリサイズ $m_c \in \mathbf{R}$
- 目的関数 $F : (M \mapsto \{0,1\}) \mapsto \mathbf{R}$
- クライアント側のメモリ容量 $m \in \mathbf{N}$
- ユーザインタフェースを含むモジュールなど，クライアント側に配置しなければならないモジュールの集合 $M_c \subset M$
- データベースにアクセスするモジュールなど，サーバ側に配置しなければならないモジュール $M_s \subset M$
- モジュール間の通信量 $T : M \times M \mapsto \mathbf{N}$
- モジュール間の呼び出し回数 $C : M \times M \mapsto \mathbf{N}$
- 各モジュールの実行時間 $Q \mapsto \mathbf{N}$
- 目的関数の係数 $K_1, K_2, K_3 \in \mathbf{R}$

ここで，モジュールの配置 $D : M \mapsto \{0,1\}$ あるモジュール $m \in M$ が端末に配置される場合には $D(m) = 0$，サーバへ配置される際には $D(m) = 1$ となるような関数とする．クライアント側に属するモジュールサイズの合計がクライアント側のメモリ容量以下である必要があるため，制約条件は，

$$\sum_{m:D(m)=0} S(m) < m_c \tag{1}$$

となる．また，目的関数 F は任意のモジュールの端末，サーバへの割り当て D に対してその割り当ての評価値を返す関数として以下のように定める．

$$\begin{aligned} F(D) = {} & K_1 \sum_{m_1,m_2:D(m_1)\neq D(m_2)} T(m_1,m_2) \\ & + K_2 \sum_{m_1,m_2:D(m_1)\neq D(m_2)} C(m_1,m_2) \\ & + K_3 \sum_{m:D(m)=0} Q(m) \end{aligned} \tag{2}$$

ここで，K_1, K_2, K_3 は，それぞれ通信量の係数，呼び出し回数の係数，実行時間の係数とした．通信量を最小にしたい場合には K_1，通信遅延を最小にしたい場合には K_2，携帯端末における消費電力を最小にしたい場合には K_3 の値を他より大きな値に設定すればよい．

3.3 SA アルゴリズムを用いた分割法

本章では，本研究で提案する分割手法について述べる．本研究で対象とする問題は，グラフ分割問題 [10] から多項式時間で帰着可能であるため NP 困難

表 1: モジュール数 30 個の場合の S/C 間総通信量

ループ回数	50	100	150
SA1	136KB (0.28s)	136KB (0.57s)	98KB (0.85s)
SA2	130KB (0.29s)	104KB (0.56s)	104KB (0.85s)
総検索	98KB(200.54s)		

問題である．よって，本問題は実用的な時間で最適解を求めることが困難と考えられるため，ヒューリスティックアルゴリズムを用いて近似解を求めることにする．ここでは時間をかけることで最適解に非常に近い解が得られると言われる SA(Simulated Annealing) 法による近似解法を用いた．

SA 法では，解の近傍を探索し，解が改善されればそれに置き換えるが，解が改善されない場合でもある確率により置き換えを行う．ある解候補 D から，近傍解 D' をランダムに生成し，評価値の差 $\Delta C = F(D') - F(D)$ を計算する．$\Delta C < 0$ である場合は，作成した近傍解を解候補として記録し，そうでない場合でも確率 $\exp(-\Delta C/T)$ で解候補とする．ここで，T を温度パラメータと呼び，近傍解探索を一定回数行う毎に $T_{k+1} = k\,T_k$ (k:定数 (冷却係数)) という式に従って減少させる．

ここでは以下の 2 通りの初期解生成法を実装し，性能の比較を行った．

- (SA1):クライアントのメモリ制限内でモジュールを生成した順にクライアント側に配置していき，クライアント側に入りきらない残りのモジュールはサーバ側にすべて配置する．
- (SA2):クライアント側に固定されるモジュールとの通信量が多いモジュールから順にクライアント側のメモリ制限内で配置する．

また，近傍解の生成は以下の方針で行った．

- そのモジュールがクライアント側にある場合，サーバ側に移動．
- そのモジュールがサーバ側にある場合，それをクライアント側に移し，クライアント側のモジュールサイズの合計がメモリ容量以下である場合はそれを配置候補とする．もし，クライアント側のメモリ容量より大きくなっていれば，再びモジュールをランダムに 1 つ選択し，同じ作業を繰り返す．

表 2: モジュール数 50 個の場合の S/C 間総通信量

ループ回数	500	1000	2000
SA1	251KB (7.08s)	240KB (14.14s)	213KB (28.29s)
SA2	253KB (7.08s)	253KB (14.13s)	213KB (28.31s)
総検索	—(—)		

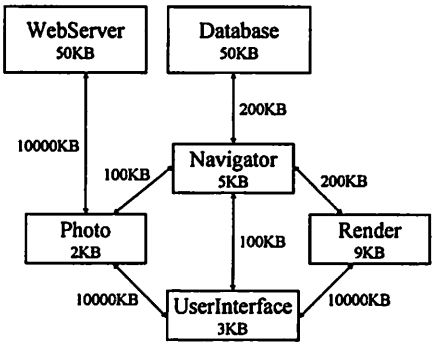


図 1: SA 手法評価例題

3.4 例題アプリケーションによる SA 手法の評価

提案実装手法の有用性を評価するため，例題を用いた実験を行った．まず，ランダムに生成したモジュールデータを用いて通信量を最小化するような分割を求め，SA 手法の性能を評価した．モジュール数 30 個の場合，総探索で得た最適解と同様の解が非常に短い時間で得られた (表 1)．また，モジュール数を 50 個にした場合には，総探索では現実的な時間で解が得られなかったが，SA を用いた手法ではループ回数を増やすことで性能の良い解が得られた (表 2)．

次に，携帯端末のリソース制限を変えて通信量が最小となるよう分割を行った場合の評価を行った．実験は，提案システムを図 1 のように実装し，シミュレーション結果が仮に表 3 となったと仮定して提案手法により分割を行った．ここでは写真情報や，ベクトルグラフィックレンダリングモジュール (Render) の出力などは大きなサイズになるものとして例題を作成した．この情報を用いて，UserInterface をクライアント側に，WebServer と Database をサーバに固定し，初期配置は SA2 を適用し提案手法を用いて分割を行った．また，クライアント側のメモリサイズの設定をいくつかの値に変えることでどのように結果が変化するかを調べた．その結果，表 4 に示したような分割結果が得られた．実験

表 3: 例題 Java アプリケーション

	クラス名	サイズ	(通信相手, 通信量 (KB))
1	UserInterface	3KB	(2, 100), (3, 10000), (4, 10000)
2	Navigator	5KB	(1, 100), (3, 100), (4,200), (6, 200)
3	Photo	2KB	(1, 10000), (2, 100), (5, 10000)
4	Render	9KB	(1, 10000), (2, 200)
5	WebServer	50KB	(3, 10000)
6	Database	50KB	(2, 200)

表 4: 提案手法による分割の結果

メモリ (KB)	クライアント側	サーバ側	通信量 (KB)
10	1	2,3,4,5,6	20100
15	1,4	2,3,5,6	10300
20	1,2,3,4	5,6	10200

結果から，メモリサイズを大きくすると分割の自由度が高くなる分，クライアント・サーバ間の通信量がより小さくなるような配置になっていることが分かる．

4 提案システムの使用例

4.1 ナビゲーションシナリオ作成例

提案システムを実装し，実際にナビゲーションシナリオを作成した．ここでは，KDDI の提供する Ez アプリ環境を対象として Java 言語を用いて実装を行った．ただし，公開されているアプリケーション実行環境のセキュリティー上の制限から写真データを Java のアプリケーションによりサーバに転送する事が出来なかったため，写真は別に手動で登録するという実装とした．なお以下の実行画面は携帯電話エミュレータにより実験を再現した物である．

まず，ナビゲーションの出発地に行き作成システムを起動する (図 2(a))．位置情報，コメントをサーバに登録する (図 2(b)，(c))．本来，その際にシステム上で写真情報も登録すべきであるが，上記制限から，ここでは後で手動で写真ファイルをアップロードする必要がある．この手順を目的地まで繰り返すことで，ナビゲーションシナリオを作成する．

4.2 ナビゲーションシナリオ例

前節の手順により表 3 のようなデータがデータベースに登録される．各地点に対して，その座標とその地点の写真及び次の地点が示されており，これらを利用することで，ナビゲーションシナリオに沿って利用者を誘導することが出来る．

4.3 ナビゲーションシナリオ参照例

作成したナビゲーションシナリオを参照する場合には，まず参照システムを起動し，位置情報を取得する (図 2(e),(f))．その情報を元にサーバに問い合わせを行うことで写真を取得できる (図 2(g))．写真情報はメニューから次の地点を選択することで順に取得でき，目的地までのナビゲーションを受けることが出来る (図 2(h))．

5 まとめ

本研究では，位置情報と写真を用いたパーソナルナビゲーションシステムの提案を行った．また，実装はシステムを最適に分割し，クライアントとサーバに割り当てる手法で行う．例題に適用して分割手法の性能評価を行った．その結果，SA を用いた方法は提案システムのようなアプリケーションに適していることが分かった．

今後の課題としては，ナビゲーションシステム全体の端末へのシミュレーション手法の適用，実装実験を行う．また，ナビゲーションシステムに登録された写真のコンテキスト情報に基づく分類，提示法や，サーバシステムのより応用的な実装を考えている．現在は位置情報に基づいて付近の写真を取得するのみとしているが，付与されたコメントを元に最も必要な写真が得られるシステムにしたいと考えている．また，今回のシステムを発展させることで，位置情報利用ミドルウェアとして，写真のみならず動画等も利用できるようなフレームワークの提案を行いたい．

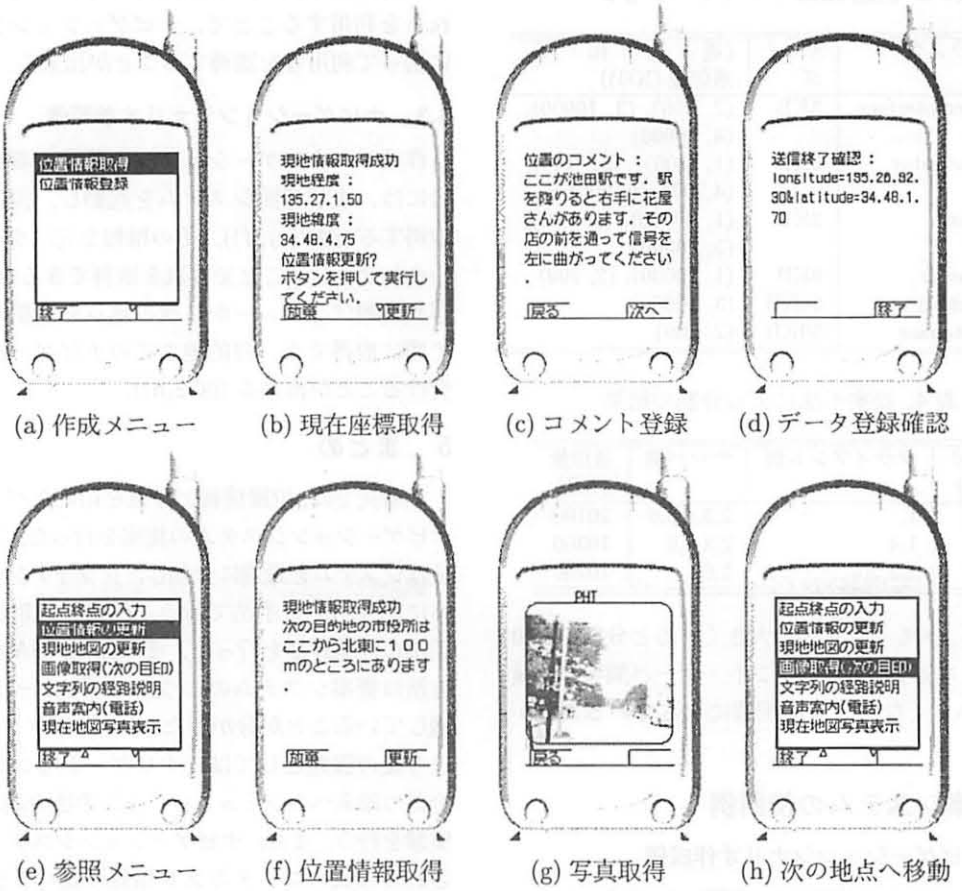
参考文献

[1] Grosso W. : Java RMI, オライリー・ジャパン (2002)

[2] HORB Open : HORB Open, http://www.horbopen.org/ (2001)

[3] Matjaz B.J., Ivan R., Marjan H., Alan P.S. and Simon N. : Java 2 Distributed Object Models Performance Analysis, Comparison and Optimization, *Proceedings of 7th International Conference on Parallel and Distributed Systems (ICPADS'00)*, pp. 239–246 (2000)

[4] Kalogeraki V., Melliar-Smith P.M. and Moser L.E. : Using Multiple Feedback Loops for Object Profiling, Scheduling and Migration in Soft Real-Time Distributed Object Systems,



(a) 作成メニュー (b) 現在座標取得 (c) コメント登録 (d) データ登録確認

(e) 参照メニュー (f) 位置情報取得 (g) 写真取得 (h) 次の地点へ移動

図 2: ナビゲーションシナリオ作成・参照

point No.	longitude	latitude	現地写真	次のポイントの写真
始点	135.27.1.50	34.48.4.75	image1.png	image2.png
2	135.28.37.75	34.48.21.25	image2.png	image3.png
3	135.27.1.30	34.48.1.75	image3.png	image4.png
4	135.26.32.30	34.48.1.70	image4.png	image5.png
終点	135.26.30.1	34.48.1.20	image5.png	

図 3: ナビゲーションシナリオ例

Proceedings of 2nd IEEE International Symposium on Object-Oriented Real-Time Distributed Computing, pp. 291–300 (1999)

[5] Flores A.P., Nacul A., Silva L., Netto J., Pereira C.E. and Bacellar L. : Quantitative Evaluation of Distributed Object-Oriented Programming Environments for Real-Time Applications, *Proceedings of 2nd IEEE International Symposium on Object-Oriented Real-Time Distributed Computing*, pp. 133–138 (1999)

[6] Kono K. and Masuda T. : Efficient RMI, Dynamic Specialization of Object Serialization, *Proceedings of The 20th International Conference on Distributed Computing Systems (ICDCS 2000)*, pp. 308–315 (2000)

[7] Zhao J. : Slicing Concurrent Java Programs, *7th International Workshop on Program Comprehension*, pp. 126–133 (1999)

[8] Zhao J. : Multithreaded Dependence Graphs for Concurrent Java Program, *International Symposium on Software Engineering for Parallel and Distributed Systems*, pp. 13–23 (1999)

[9] Sait S.M.,Youssef H. : 組合せ最適化アルゴリズムの最新手法基礎から工学応用まで, 白石 洋一訳, 丸善 (株) 出版事業部 (2002)

[10] Garey M.R. and Johnson D.S. : Computers and intractability A guide to the theory of NP-completeness, Freeman (1979)