一般社団法人　電子情報通信学会
THE INSTITUTE OF ELECTRONICS,
INFORMATION AND COMMUNICATION ENGINEERS

信学技報
IEICE Technical Report
CPSY2018-118,DC2018-100 (2019-03)

拡張現実処理のオフローディングにおけるサーバの割り当てに関する考察

竹内　裕紀† 　酒井　和哉† 　福本　聡†

† 首都大学東京大学院 システムデザイン研究科 電子情報システム工学域 〒191–0065 東京都日野市旭ヶ丘 6–6
E-mail: †takeuchi-yuuki@ed.tmu.ac.jp, ††{ksakai,s-fuku}@tmu.ac.jp

あらまし　近年，AR(Augumented Riality, 拡張現実感) を利用した，ゲームや仕事効率化のためのアプリケーションソフトウェア，ナビゲーションなど様々な応用が急増している．AR のサービスを利用するには，ユーザが撮影した画像データからオブジェクトを検知・認識するためのアルゴリズムを適用する必要がある．精度の高いサービスをユーザが利用するには画像データの画質を上げ，より大きな計算量のアルゴリズムを適用させる必要がある．そのため，データの処理時間やサーバへのデータ転送による遅延時間が大きい．伝送遅延を抑えるために，ユーザ自身の端末でデータの処理を行うことも考えられるが，これらのアプリケーションはユーザのモバイル端末で利用されることが想定されるため，端末で消費できる電力量には制限がある．したがって，消費電力量と遅延時間の双方を戦略的に抑えることが要求される．本研究では，最適化問題を応用したサーバ割り当てによってこれを実現できることを示す．

キーワード　拡張現実，サーバ割り当て，オフローディング

Yuki TAKEUCHI†, Kazuya SAKAI†, and Satoshi FUKUMOTO†

† Faculty of Systems Design, Department of Electrical Engineering and Computer Science, Tokyo Metropolitan University, 6–6 Asahigaoka, Hino-shi, Tokyo 191–0065, Japan
E-mail: †takeuchi-yuuki@ed.tmu.ac.jp, ††{ksakai,s-fuku}@tmu.ac.jp

1. まえがき

近年，AR (Augmented Reality，拡張現実感) [1]〜[3] を応用したサービスが増えている．その市場は，2022 年までに 850 億ドルから 900 億ドルにまで成長すると言われている [4]．AR とは，現実世界においてディスプレイなどに仮想的なオブジェクトを表示することでユーザの世界を拡張する技術である (図 1)．例えば，ポケモン GO(注1)というゲームでは，街中などの風景に溶け込んで仮想的にポケモンと呼ばれるオブジェクトが出現する．単純にディスプレイにこのポケモンが表示されるだけでなく，ディスプレイに表示されている物体を認識し，その上にポケモンが乗ったり移動したりする．この技術によって，ユーザはあたかも現実世界にポケモンが登場したような体験ができる．また，Google が提供する翻訳サービス(注2)では，スマートフォンのディスプレイに表示されている言語を認識し，他の国の言語に翻訳されたものに書き換えるものがある．現実世界の文章を仮想的に書き換えることで，直感的に翻訳サービスを利用することができる．さらに，自動車にも AR 技術が応用される．例えば，フロントガラスにユーザが進むべき方向を表示して目的地までナビゲーションするシステムがある [5]．一般的なカーナビゲージョンシステムでは，案内表示を見るためにユーザは車に備え付けられた画面に視線を移さなければならないが，このシステムでは前方を見ながら進路を確認できるため，交通事故の減少につながると期待されている．

AR を応用したこれらのサービスは，ユーザとサーバとで大きなサイズのデータをやり取りする必要がある．AR アプリケーションはユーザが撮影した画像データをもとにして，現実世界のオブジェクトを認識・検知する (図 2)．しかし，画像データはサイズが大きく転送に時間がかかるため，サーバが画像データを処理をする際にはどうしても伝送遅延が生じる．また，伝送遅延を少なくするためにユーザ自身の端末で画像データの処理を行うと，大量の計算が必要になり，大きな電力量消費が発生する．AR 技術は，スマートフォンなどの携帯端

(注1)：https://www.pokemongo.jp/
(注2)：https://itunes.apple.com//jp/app/google-%E7%BF%BB%E8%A8%B3id414706506?mt=8

図 1: AR アプリケーションの利用例．(左)：街中に登場するゲームキャラクター (https://www.pokemongo.jp/howto/play/images/cap_01.jpg)，(中央)：日本語に翻訳される英語 (https://vrinside.jp/wp-content/uploads/g1.jpg)，(右)：フロントガラスに表示される情報 (https://response.jp/imgs/thumb_h2/1143719.jpg)．

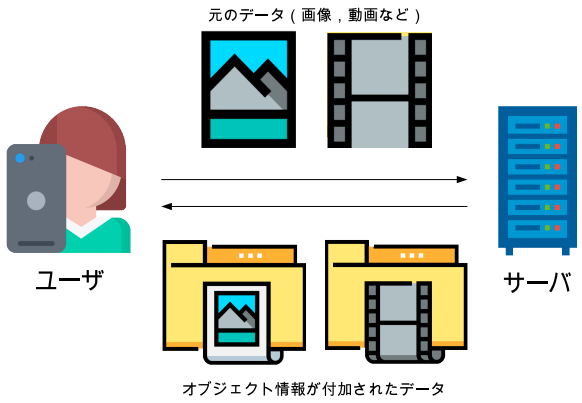

図 2: AR アプリケーションのシステム．

末で使用される例が多く，バッテリーの容量が小さい端末で消費電力量が大きくなると，ユーザは頻繁に端末を充電する必要にかられる．サーバとの適切な通信によって，遅延時間と消費電力量の両方を戦略的に抑えることを目指す必要がある．

本研究では，ユーザに生じる遅延時間と消費電力量を定式化し，ユーザとサーバとの割り当ての有無を変数とした最適化問題を定義する．この最適化問題から得られたサーバ割り当てによって，遅延時間と消費電力量を効果的に抑えられることを示す．

本稿は以下のように構成されている．2. で関連研究について述べたのちに，3. で遅延時間と消費電力量について定式化する．4. で最適化問題を定義し，それを解くためのアルゴリズムを示す．その後，5. でシミュレーションについて述べる．6. は本稿のまとめである．

2. 関 連 研 究

はじめに，サーバ割り当てに関する研究として，Webb らのアルゴリズムについて述べる [6]．このアルゴリズムはオンラインゲームにおいて，遅延によって不利になるユーザが少なくなるようにサーバを割り当てることを目的としている．通信頻度が高いオンラインゲームでは，ゲーム内の環境の変化がユーザごとに異なるタイミングで反映されることがある．その変化が早く反映されたユーザに比べて，遅れて反映されたユーザはゲーム内の環境の変化に対応するタイミングが遅れるため不利になる．Webb らは効率よくサーバを割り当てることやユーザごとにデータを送信するタイミングを調整することで，このような状況を回避するアルゴリズムを提案している．

次に，AR ユーザ向けのサーバ割り当てに関連する研究として Liu らの研究 [7] について述べる．Liu らは，ユーザに生じる遅延時間を抑えつつ，オブジェクトの認識や検知について一定の精度を保証するアルゴリズムを提案している．オブジェクトの認識や検知を高精度で行うには，YOLO [8] や SSD [9] などの計算量の大きいオブジェクトの認識・検知のためのアルゴリズムが必要になる．計算能力が制限されたモバイル端末で計算量の大きいこれらのアルゴリズムを実行することは，計算のための遅延時間が大きくなることにつながる．それを抑えるためにサーバに計算を委託することが考えられるが，ユーザからサーバへの画像データなどの送信と，サーバからの処理結果の受信のための遅延時間が発生する．Liu らは，ユーザの通信状況に応じて，計算量の異なるアルゴリズムを選択してオブジェクトの認識や検知を行うことで，一定の精度を保証しつつ遅延時間を抑えらることを示している．

また，サーバを効率よく使用するために，ユーザがサーバに送信するべき画像の解像度と，サーバが受信したデータに適用させるアルゴリズムを指定する手法もある [10]．この手法では，クラウドサーバ [11] ではなくエッジサーバ [12] に計算させる状況を想定し，オブジェクトの認識率とユーザに生じる遅延時間を定式化して最適化問題を定義している．エッジサーバはクラウドサーバと違い，計算能力が著しく高いわけではないが，無線のアクセスポイントに近い場所に数多く設置され

ているという特徴をもつ．クラウドサーバと比較してデータの転送に時間がかからないため，IOT(Internet of Things) などへの活用も期待されている．

3. 定 式 化

本節では，ユーザが AR システムのサービスを受けるために生じる遅延時間と，それぞれのユーザの端末における消費電力量の定式化について述べたのちに，最適化問題として解くための提案手法について述べる．

3.1 遅延時間の定式化

ユーザの集合 U，サーバの集合 S で構成されるネットワークを考える．ユーザは WiFi や LTE などを利用しサーバと通信を行い，AR のサービスに必要な処理を行う．ユーザ u に生じる遅延時間 D_u を次のように定義する．

$$D_u = x_{u,s}(2C_{u,s} + L_u^{(s)}) + (1 - x_{u,s})L_u^{(u)}. \tag{1}$$

ここで，$C_{u,s}$，$L_u^{(s)}$ は，それぞれ，ユーザ u に割り当てられているサーバ s にデータを伝送するのにかかる伝送遅延時間，サーバ s が受信したデータを処理するのにかかる計算遅延時間である．ユーザは画像データをサーバに送信し，その画像に存在するオブジェクトの認識や識別の処理を委託する．また，$L_u^{(u)}$ はユーザが自身でデータを処理するのにかかる時間である．$x_{u,s} \in \{0,1\}$ は，ユーザ u がサーバ s に割り当てられていれば 1，割り当てられていなければ 0 となる変数である．

サーバに計算を委託する場合，まず，ユーザは割り当てられているサーバに必要なデータを送信する．サーバはデータを受信すると必要な処理を実行し，計算結果をユーザに返す．ユーザは受信した計算結果から，AR アプリケーションによるサービスを受ける．そのため，データの伝送遅延時間 $C_{u,s}$ と，サーバによる計算遅延時間 $L_u^{(s)}$ の 2 種類の遅延時間がユーザ u に生じる．ユーザ u がサーバ s に計算を委託するデータの大きさを w_u としたとき，$C_{u,s}$，$L_u^{(s)}$ をそれぞれ次のように記述する．

$$C_{u,s} = \frac{w_u}{b_{u,s}}. \tag{2}$$

$$L_u^{(s)} = \frac{w_u}{F_s / \sum_{u \in U} x_{u,s}}. \tag{3}$$

$b_{u,s}$ は，ユーザ u からサーバ s への通信路の帯域幅を表し，F_s はサーバ s の計算能力を表す．複数のユーザに割り当てられているサーバは，それぞれのユーザに対して等しく計算能力を提供することを前提としている．

また，ユーザの端末で計算が実行される場合，データの伝送遅延時間は発生しないが，サーバよりも小さい計算能力でデータを処理する必要がある．$L_u^{(u)}$ を次のように記述する．

$$L_u^{(u)} = \frac{w_u}{f_u}. \tag{4}$$

サーバの計算遅延時間と同様に，f_u はユーザ u がもつ計算能力を表す．

3.2 消費電力量の定式化

ユーザが自身でデータを処理する際やデータを伝送する際にはユーザの端末は電力量を消費する．ユーザ u の端末が消費する電力量 P_u を次のように定義する．

$$P_u = x_{u,s}P_u^{(s)} + (1 - x_{u,s})P_u^{(u)}. \tag{5}$$

$P_u^{(s)}$，$P_u^{(u)}$ はそれぞれ，ユーザ u が割り当てられているサーバ s にデータを送受信するときに消費する電力量，ユーザ u が自身でデータを処理するときに消費する電力量である．以下では，ユーザがデータを伝送する際に生じる消費電力量を伝送消費電力量，ユーザがデータを処理する際に生じる消費電力量を計算消費電力量と呼ぶことにする．

先行研究 [13], [14] と同様に，伝送消費電力量は送受信するデータの大きさに比例するとすれば，ユーザ u からサーバ s へ大きさ w_u のデータを送信するために必要な伝送消費電力量は，次のように記述できる．

$$P_u^{(s)} = \beta w_u. \tag{6}$$

β は単位データ当たりの伝送消費電力量を表すパラメータである．ユーザの端末の種類によって異なるが，ここでは，先行研究と同様に $10^{-9} \leq \beta \leq 10^{-7}$ を想定する．

また，先行研究 [15] により，ユーザ u の端末の計算消費電力量 $P_u^{(u)}$ は，ユーザが持つ計算能力 f_u に依存する．$P_u^{(u)}$ を次のように定式化する．

$$P_u^{(u)} = \alpha(f_u)^\gamma \tag{7}$$

α，γ は，アーキテクチャの性能に依存するモデルパラメータであり，ここでは，先行研究と同様に $\alpha = 10^{-11}$，$2 \leq \gamma \leq 3$ を想定する．

以上の定義から，システム内の各ユーザに生じる遅延時間の総和 D は次のように求まる．

$$\begin{aligned} D &= \sum_{s \in S} \sum_{u \in U} D_u \\ &= \sum_{s \in S} \sum_{u \in U} \left[x_{u,s} \left(\frac{2w_u}{b_{u,s}} + \frac{w_u \sum_{u \in U} x_{u,s}}{F_s} - \frac{w_u}{f_u} \right) + \frac{w_u}{f_u} \right]. \end{aligned} \tag{8}$$

また，システム内の各ユーザに生じる消費電力量の総和 P は，

$$\begin{aligned} P &= \sum_{s \in S} \sum_{u \in U} P_u \\ &= \sum_{s \in S} \sum_{u \in U} \left[x_{u,s} \left\{ \beta w_u - \alpha(f_u)^\gamma \right\} + \alpha(f_u)^\gamma \right] \end{aligned} \tag{9}$$

と求まる．

4. 提案手法の概要

遅延時間と消費電力量を戦略的に抑えたサーバの割り当てを求めるために，$x_{u,s}$ を変数とした割り当て問題を定義する．遅延時間と消費電力量はトレードオフの関係にある．消費電力量を抑えるために全ての処理をサーバに委託すると，データの転送に時間がかかるため遅延時間が大きくなる．逆に，遅

延時間を抑えるために全てのデータをユーザ自身で処理すると，ユーザの計算資源を多く使うことになり，消費電力量が大きくなる．適切な解 $x_{u,s}$ を求めるために次の割り当て問題を定義する．

$$\mathcal{P}_0 : \min_{u\in U, s\in S} \quad F_0 = \theta \cdot Scale(D) + (1-\theta)\cdot Scale(P)$$

subject to

$$C_1 : \quad \sum_{s\in S} x_{u,s} = 1 \quad (\forall u \in U)$$

$$C_2 : \quad x_{u,s} \in \{0,1\} \quad (\forall u \in U, \forall s \in S)$$

$Scale : \mathcal{R} \to [0,1]$ は関数の値域を 0 から 1 の範囲に正規化して無次元化する関数である．$\theta \in [0,1]$ は遅延時間と消費電力量のどちらに重みを置くかを示すパラメータである．制約式 C_1, C_2 は，各ユーザにただ一つのサーバが割り当てられることを示す．この問題 $\mathcal{P}_0$ は，変数 $x_{u,s}$ がバイナリであるため，0-1 整数計画問題となる．

0-1 整数計画問題は NP 困難であることが示されており，$\mathcal{P}_0$ の実行可能な解の組み合わせは $(2^{|U|\cdot|S|})$ 通りあるため最適解を多項式時間で求めることは困難である，そこで，$x_{u,s}$ を実数変数 $\hat{x}_{u,s}(0 \le \hat{x}_{u,s} \le 1)$ に置き換えた以下の緩和問題 $\mathcal{P}_1$ を考える．

$$\mathcal{P}_1 : \min_{u\in U, s\in S} \quad F_1 = \theta \cdot Scale(D) + (1-\theta)\cdot Scale(P)$$

subject to

$$C_1' : \quad \sum_{s\in S} \hat{x}_{u,s} = 1 \quad (\forall u \in U)$$

$$C_2' : \quad 0 \le \hat{x}_{u,s} \le 1 \quad (\forall u \in U, \forall s \in S)$$

問題 $\mathcal{P}_1$ では，バイナリ変数を 0 から 1 の連続値に緩和する．ここで，目的関数 F_1 の二階偏微分は，

$$\frac{\partial^2 F_1}{\partial \hat{x}_{i,j}\partial \hat{x}_{u,s}} = \begin{cases} \frac{2w_i}{F_j} & (i = u \quad and \quad j = s) \\ 0 & (i \neq u \quad or \quad j \neq s) \end{cases} \tag{10}$$

となる．したがって，目的関数 F_1 は凸関数である．そして，制約式も明らかに凸関数であるから，問題 $\mathcal{P}_1$ は凸最適化問題である．凸最適化問題の局所的な最小値は大域的な最小値と一致する．そのため，問題 $\mathcal{P}_1$ の最適解は比較的容易に求められる．本稿では，問題 $\mathcal{P}_1$ を内点法 [16] を用いて解く．内点法は高速に厳密解へと収束することが知られている．

その後，$\mathcal{P}_1$ の最適解から各ユーザにどのサーバを割り当てれば良いかを決定する．$\mathcal{P}_1$ の最適解の値 $\hat{x}_{u,s}$ は，サーバ s がユーザ u に割り当てられることの適切さを表す．次式からユーザ u が割り当てられるサーバ s を求める．

$$x_{u,s} = \begin{cases} 1, & \text{if } s = \underset{j\in S}{\operatorname{argmax}}\, \hat{x}_{u,j} \\ 0, & otherwise \end{cases} (\forall u \in U). \tag{11}$$

5. シミュレーション

この節では，本研究で行ったシミュレーションについて述べる．最初に，複数のユーザとサーバからなるネットワークを構成した．各ユーザは，それぞれ，アクセスポイントを経由して高々一つのサーバに接続される．各サーバは，一つ以上のサーバと接続されているとする．そして，各ユーザにただ一つのサーバを割り当てることにする．このとき割り当てられるサーバはユーザが経由するアクセスポイントに接続されているサーバと同じとは限らない．また，ユーザはデータの処理をサーバに委託，あるいは，自身で処理することを想定した．データの処理をサーバに委託する場合，ユーザは，処理が必要なデータを割り当てられているサーバへ送信する．割り当てられているサーバとアクセスポイントを経由して接続しているサーバが異なる場合は，サーバを経由しながら，目的となるサーバへデータを転送する．そのユーザに割り当てられているサーバがユーザからデータを受信すると，そのサーバは必要な処理を開始する．処理が完了したのちに，サーバはデータの送信元のユーザへ処理結果を転送する．また，ユーザ自身でデータを処理する場合は，データを転送せずにユーザ自身の計算資源を消費してデータの処理を行う．処理が必要なデータはランダムなタイミングで生成される．ユーザは，そのユーザに割り当てられているサーバがあればそのサーバにデータを送信し，なければ自身でデータを処理する．ユーザが AR サービスを 10000 秒間利用するケースに相当するシミュレーションをユーザ数やサーバ数を増やしながら，表 1 に示した手法を実装してそれぞれ 100 回ずつ行った．また，シミュレーションにあたって設定したパラメータの値は，先行研究に倣っている．パラメータの一覧を表 2 に示す．

表 1: 実装した手法一覧

手法	概要
Nearest	ユーザに対して最も近いサーバを割り当てる
AllClient	全てのデータをユーザ自身で処理する
Random	ランダムにサーバを割り当てる
Proposed	提案手法

表 2: パラメータとその値の一覧

パラメータ	値
α	10^{-11}
β	10^{-8}
γ	2.5
w_u	100 ∼ 1000[kB]
$b(u,s)$	10 ∼ 20[Mbps](下り) 2 ∼ 5[Mbps](上り)

図 3 は，それぞれ，サーバ数を 10 に固定してユーザ数を増やしたときの各ユーザに生じる遅延時間，消費電力量の平均値のグラフである．また，このときパラメータ $\theta = 0.5$ である．ユーザ自身でデータの処理をする場合に生じる遅延時間は最も小さくなるが，消費電力量は大きくなる．また，各ユーザに最も近いサーバを割り当てた場合は，消費電力量が小さい代わりに大きな遅延時間が生じる．しかし，サーバ割り当てを最適化した場合，最も近いサーバをユーザに割り当てたときよりも

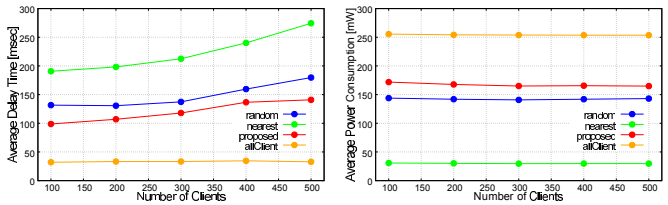

図 3: ユーザ数を増加させたときの遅延時間と消費電力量 ($a = 0.5$).

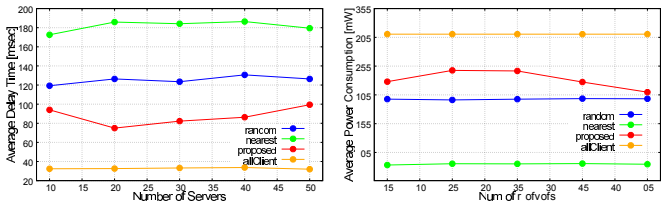

図 4: サーバ数を増加させたときの遅延時間と消費電力量 ($a = 0.5$).

遅延時間が小さく，全ての処理を自身で行うときよりも消費電力量を小さく抑えることができる．また，ユーザの数を増やすと，ユーザに生じる遅延時間が大きくなる．これは各サーバが処理するべきデータの量が増えたことによる遅延である．

図 4 は，それぞれ，ユーザ数を 100 に固定してサーバ数を増やしたときの各ユーザに生じる遅延時間，消費電力量の平均値のグラフである．上記と同様に，$\theta = 0.5$ である．提案手法でユーザにサーバを割り当てたときにユーザに生じる遅延時間は，ユーザに最も近いサーバを割り当てたときの約 50% に抑えられている．また，提案手法を用いてユーザにサーバを割り当てたときにユーザに生じる消費電力量は，全てのデータをユーザ自身で処理するときの約 60% に抑えられている．

パラメータ θ の値を 0 から 1 に変えながらシミュレーションしたときの，ユーザに生じる遅延時間と消費電力量の平均値を図 5 に示す．このときのサーバ数は 10 である．パラメータを 0 から 1 に増加させると遅延時間は減少する傾向にあり，ユーザの端末に生じる消費電力量は増加する．これは，パラメータ θ は遅延時間と消費電力量のどちらに重みを置くかを表すため当然の結果である．θ が 0.75 より小さい場合は，それぞれの遅延時間と消費電力量に大きな変化はない．0.75 よりも多くなるとこれらの変化は大きくなる．これは θ が小さいことは，消費電力量を抑えることに重みを置いて割り当てを最適化させることを示す．また，θ の値が大きいとき，つまり，ユーザに生じる遅延時間に重みを置いた場合は，各ユーザのサーバへの通信路の状況に応じてサーバ割り当てを最適化させている．それぞれの通信路の帯域幅は，ランダムに変化させため，短い遅延時間でデータを伝送できるユーザとそうでないユーザの差が現れる．そのため，短い遅延時間でデータを伝送できるユーザにはサーバを割り当てて，そうでないユーザは自身で処理することで，システム内の各ユーザに生じる遅延時間を短くする割り当てが求められる．その結果，自身でデータを処理するユーザが増えて，システム内の各ユーザに生じる消費電力量の総和が増えたと考えられる．

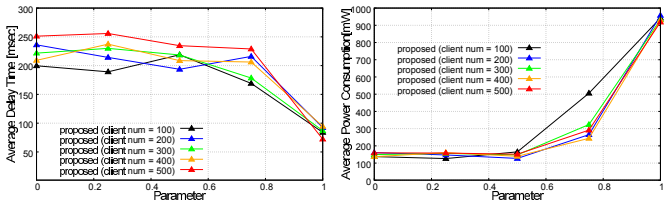

図 5: パラメータ a を変化させたときの遅延時間と消費電力量．

6. ま と め

本研究では，AR アプリケーションを利用するユーザに生じる遅延時間と消費電力量を定式化して最適化問題を定義した．この問題の最適解に応じてサーバにユーザを割り当てることで，ユーザの端末の消費電力量と遅延時間を戦略的に抑えられることを示した．今後の課題としては，実機での検証などがあげられる．

文 献

[1] D. Chatzopoulos, C. Bermejo, Z. Huang, and P. Hui, “Mobile Augmented Reality Survey: From Where We Are to Where We Go,” IEEE Access, vol.5, pp.6917–6950, 2017.

[2] D. Wagner and D. Schmalstieg, “Making Augmented Reality Practical on Mobile Phones, Part 1,” IEEE Compt. Graph. and Appl., vol.29, no.3, pp.12–15, May 2009.

[3] D. Wagner and D. Schmalstieg, “Making Augmented Reality Practical on Mobile Phones, Part 2,” IEEE Comput. Graph. Appl., vol.29, no.4, pp.6–9, July 2009.

[4] Digi-Capital, “Ubiquitous $90 billion AR to dominate focused $15 billion VR by 2022 — Digi Capital,” Accessed 2019-1-28. https://www.digi-capital.com/news/2018/01/ubiquitous-90-billion-ar-to-dominate-focused-15-billion-vr-by-2022/

[5] K. Lebeck, T. Kohno, and F. Roesner, “How to Safely Augment Reality: Challenges and Directions,” Proc. of the 17th Int. Workshop on Mobile Compt Syst. and Appl., pp.45–50, HotMobile ’16, ACM, New York, NY, USA, 2016.

[6] S.D. Webb and S. Soh, “Adaptive client to mirrored-server assignment for massively multiplayer online games,” Multimedia Compt. and Netw. 2008, vol.6818, p.681801, 2008.

[7] Q. Liu, S. Huang, J. Opadere, and T. Han, “An Edge Network Orchestrator for Mobile Augmented Reality,” INFOCOM, pp.756–764, April 2018.

[8] J. Redmon and A. Farhadi, “YOLO9000: Better, Faster, Stronger,” CVPR, pp.6517–6525, July 2017.

[9] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A.C. Berg, “SSD: Single Shot Multibox Detector,” ECCV, pp.21–37, 2016.

[10] Q. Liu and T. Han, “DARE: Dynamic Adaptive Mobile Augmented Reality with Edge Computing,” ICNP, pp.1–11, 2018.

[11] T. Verbelen, P. Simoens, F. De Turck, and B. Dhoedt, “Cloudlets: Bringing the Cloud to the Mobile User,” MCSACM, pp.29–36 2012.

[12] M. Satyanarayanan, “The Emergence of Edge Computing,” Compt., vol.50, no.1, pp.30–39, 2017.

[13] Y. Xiao, R. Bhaumik, Z. Yang, M. Siekkinen, P. Savolainen, and A. Yla-Jaaski, “A System-Level Model for Runtime Power Estimation on Mobile Devices,” GREENCOM-CPSCOM, pp.27–34, Dec. 2010.

[14] B. Goel, S.A. McKee, R. Gioiosa, K. Singh, M. Bhadauria, and M. Cesati, “Portable, Scalable, per-Core Power Estima-

tion for Intelligent Resource Management," GREENCOM, pp.135–146, IEEE, Aug. 2010.

[15] X. Chen, L. Jiao, W. Li, and X. Fu, "Efficient Multi-User Computation Offloading for Mobile-Edge Cloud Computing," IEEE/ACM Trans. on Netw., vol.24, no.5, pp.2795–2808, Oct. 2016.

[16] N. Karmarkar, "A new polynomial-time algorithm for linear programming," STOC, pp.302–311, 1984.