Vol.2014-DPS-160 No.2
2014/7/24

スマートフォンの電力消費量測定用 Web ブラウザの開発と評価

西村顕[†1]　荒瀬亮[†2]　杉田薫[†2]

コミュニケーションサービス全体の電力消費量の削減を目標として,オンラインビデオ再生時における QoS パラメータによるスマートフォンの電力消費量への影響を明らかにするため,電力消費量測定用 Web ブラウザの開発と評価を行った.本ブラウザは iOS 版と Android 版を開発しており,マイクロブログやソーシャルネットワークサービス(SNS)を含めた様々な Web サービス利用時の電力消費量を測定可能である.本稿では,この Web ブラウザの実装と HTTP Live Streaming を使用したビデオ再生時の電力消費量への影響について報告する.

Development of Web browsers to mecasure power consumption on Smartphone

KEN NISHIMURA[†1]　RYO ARASE[†2]　KAORU SHUGITA[†2]

In order to reduce a total power consumption of the communication services, we developed and evaluated Web browsers supporting a measurement of power consumption to find some effective QoS parameters at playing video. The Web browsers are running on iOS and Android to measure a power consumption using a micro-blog and Social Network Service(SNS) and so on. In this paper, we describe our implementations and effectiveness of power consumption at playing HTTP Live Streaming.

1. はじめに

1.1 研究背景

近年,iOS や Android を搭載したスマートフォンが急速に普及している.それに伴い,マイクロブログやソーシャルネットワークサービス(SNS)のようなコミュニケーションサービスが急速に普及している.このようなサービスでは,自分が体験した出来事や身の回りで起こった出来事を瞬時に共有することや,情報を発信することが可能である.しかし,このようなサービスは,スマートフォンの電力消費量の増加を招いており,充電器やモバイルバッテリを必需品とする人が増加する傾向にある.スマートフォンの利用可能時間を延ばすためにバッテリ容量が増加されているが,スマートフォンの利用可能時間に対する不満は解消されていないのが現状である.また,電力消費量の削減に対する関心が高くなりつつある[1].

1.2 関連研究

コンピュータネットワーク分野における電力消費量削減に関する研究としては，ネットワークの自立制御による通信路の電力消費量削減に関する研究[2]，アプリケーションプログラム参加の省電力制御に関する研究[3]，オブジェクトの監視・追跡を行う無線マルチメディアセンサネットワークの稼働時間延長および QoS 確保のためのルーティング手法[4]，多様な要求品質を持つ移動端末ユーザへのリソース効率の良いビデオ配信方式[5]，快適度の低下を最小限に抑える省エネデバイス制御手法[6]が報告されている．しかし,コミュニケーションサービス全体の電力消費量削減に関する報告例はほとんどなく,スマートフォンの電力消費量削減についてもほとんど報告がなされていない.

以上のことから,本研究ではこのクライアントとして主流となってきているスマートフォンに着目し,この電力消費量を測定するための Web ブラウザの開発を行っている.本稿では,スマートフォンの電力消費量測定用 Web ブラウザの実装と,予備実験を iOS と Android で行ったので報告をする.

2. 電力消費量測定用 Web ブラウザの開発

電力消費量測定用 Web ブラウザの機能を表 1 に示す.実験パラメータの入力では,ビデオ品質である動画のフレームレートと端末の画面輝度を設定し,測定が開始可能である.Web ページの表示は,測定開始後に実行され,この表示中は一般的な Web ページの操作が可能である.実験対象の測定は,Web ページの表示中の,任意のタイミングで実行可能である.実験結果のファイル出力は,実験中に測定対象の値を CSV ファイルとして出力可能である.

表 1　電力消費力測定用 Web ブラウザ機能

機能	入力	出力
実験パラメータ入力画面の表示	動画は30fpsと12fpsが選択可能 画面輝度は0~100%で指定可能	-
実験用画面の表示	実験対象のURL 表示されたWebページの操作	Webページの表示
実験対象の測定	-	バッテリ残量[%] 1秒間あたりのCPU使用時間[ms] 現在表示中のURL 現在の画面輝度[%]
実験結果のファイル出力	-	CSVファイル

†1 福岡工業大学大学院
Graduate School of Fukuoka Institute of Technology
†2 福岡工業大学
Fukuoka Institute of Technology

3. モジュールの構成

図 1 は本ブラウザの MVC モデルで設計したモジュール構成である.ビューは表示部分,モデルはデータ管理部分,コントローラは,ビューとモデルの管理部分である.

3.1 ビュー

表 2 にビューのモジュール構成を示す. URL と画面輝度の実験パラメータは入力後,実験が終了するまで保存しておく必要がある.ここで,入力された値はコントローラ経由でモデルに送られる.ここでは,実験パラメータ入力後,実験開始ボタンが押されると,実験画面に遷移され,パラメータがコントローラ経由でモデルに保存される.さらに,実験画面の読み込み終了後,コントローラへ再通知を行い,コントローラからモデルに実験パラメータの値を読み込む.これらの値を元に,読み込まれた Web ページで,実験が開始される.バッテリ残量が一定値以下になると,自動的に実験パラメータを入力する画面へ遷移する.

3.2 モデル

表 3 にモデルのモジュール構成を示す.実験パラメータ値はどのビューからでもアクセス可能である必要性があるため,モデルにデータとして保存される.モデルではバッテリ残量,CPU 使用時間といった,デバイス自体の様々な値を取得して記録する.

3.3 コントローラ

コントローラは主にビューとモデルからの通知に対して操作を実際に行う.ビューへの要求を表 4 に示す. また,モデル対象への要求を表 5 に示す.

最初にビューに対する要求を受け取ると実験パラメータがモデルへ保存されると同時に,実験画面へ画面が遷移し,実験が開始される.実験開始時には測定対象の値を取得と保存がバックグラウンドで実行される.実験画面へ遷移した後は,Web ページが表示される.その後は,ユーザは自由に Web ページを閲覧する事が可能となる.実験中にバッテリ残量が設定された閾値以下となった場合,実験を終了し,パラメータを入力する画面に遷移するよう通知を行う.実験パラメータ値を間違って設定しまった場合や,実験中に問題が発生した場合などには,実験を中止して元の画面へ戻るよう通知を行う.

モデルに対する主な要求は実験パラメータ入力画面で入力された値の保存である.これは主に実験画面でのパラメータを設定するために,実験パラメータ値を読み込む時に要求される.実験パラメータ入力画面で,TestStart と書かれたボタンが押されるとこの要求が通知され,実験パラメータの設定,測定の開始,測定データの CSV ファイル出力が実行される.また,実験は実験画面の操作で強制終了するか,バッテリ残量が設定された閾値以下となると要求される.この要求に従って,測定の終了処理,Web ページの初期化,Web ページのキャッシュ消去,測定データの CSV ファイルへ書き出し終了実行される.これは主にデバッグ用に実装した要求である.この要求は実験パラメータ値,測定ファイルなどの初期化と消去を行う要求であるため,実験時に押してしまうとそれまでの測定データが消去される.

表 2　ビューのモジュール構成

モジュール	入力	出力
実験パラメータ入力画面の表示	測定対象のURLが入力可能 動画は30fpsと12fpsが選択可能 画面輝度は0-100%で指定可能	画面の生成
実験用画面への遷移と表示	実験開始要求	指定したURLのページを読み込み 実験パラメータを元に画面輝度の調節

表 3　モデルのモジュール構成

モジュール	入力	出力
実験パラメータの管理	実験対象の動画データ(URL) 実験中の画面輝度[%]	実験対象の動画データ(URL) 実験中の画面輝度[%]
実験対象を任意の間隔で取得	測定開始要求	バッテリ残量を一定間隔で取得
実験結果をファイル出力	測定データ出力開始要求	CSVファイル

表 4　コントローラにおけるビューへの要求

モジュール	概要
実験開始要求	実験を開始するため以下の処理が実行 実験パラメータの保存 実験パラメータから実験画面の生成と描画
画面操作要求	画面操作に対して画面の更新を行う
実験終了要求/実験強制終了要求	実験を終了するため以下の処理が実行 測定対象のWebページに空のページの読み込みさせ初期化 実験画面の破棄を行いメモリ解放

表 5　コントローラにおけるモデルへの要求

モジュール	概要
実験パラメータの管理要求	実験パラメータをモデルに管理するよう要求
実験開始要求	実験を開始するため以下の処理が実行 実験パラメータの管理 測定対象の取得開始とCSVファイルへ出力開始
実験終了要求	実験を終了するため以下の処理が実行 測定対象の取得終了とCSVファイルへ出力終了 実験画面の破棄を行いメモリ解放
実験強制終了要求	実験終了要求と同じだが異なる点としてCSVファイルを消去する
アプリケーションリセット要求	実験パラメータは初期化を行いCSVファイルはすべて消去する

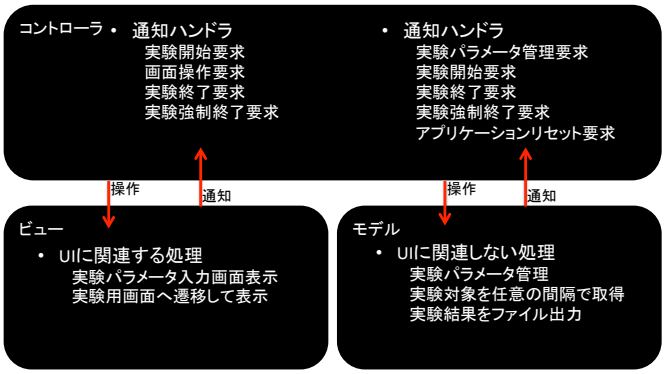

図 1　電力消費量測定用 Web ブラウザの MVC モデル

4. 実装

4.1 HTTP Live Streaming

本論文では HTTP Live Streaming(HLS)[7]で放送型のオンラインビデオを Web ページ上で HTML5 を使用し,連続再生させた時のバッテリ残量を予備実験として測定した.

4.2 iOS

4.2.1 ビュー機能

ビューは以下の実装を行った.

- 実験パラメータ入力画面の生成
- 実験開始要求による実験画面遷移
- 実験終了要求による実験画面の破棄と終了処理
- UIWebView による Web ブラウザの実装
- 測定値を画面出力する機能の実装
- 設定した画面輝度へ強制する機能の実装

実験用パラメータを設定出来るように,Storyboard と UI 用の API を利用して入力画面の実装[8]をおこなった.Storyboard では UITextView, UITextField, UIButton を利用した.これらは,文字表示のみ行うパーツ,テキストを表示と入力が可能なパーツ,実際のボタンの用に押された時にアクションが起こせるパーツである.UI 用の API を使用したオブジェクトでは,Storyboard では実装が難しく,座標の指定が必要な UIPickerView,UIToolbar,UIBarButtonItem 実装を行った.それぞれ,用意された値を選択させるパーツ,View の上で他のパーツを設置可能なパーツ,UIToolbar 上に設置出来るボタンパーツである.実験パラメータ入力画面はアプリケーション起動時の初期画面となっている.実験パラメータの値を入力して UIButton である TestStart を押すことで,実験画面へ遷移し,実験が開始される.なお,実験パラメータの値を保存するための処理については,モデルの機能説明時に行う.遷移先では,UIWebView が設置されており,Web ページの読み込みと,「戻る」「進む」「更新」「中止」といった簡単な Web ページの操作が行えるが,URL の指定はこの画面で行えない.また,Web ページをキャッシュする機能が UIWebView に実装されており無効に出来ない.しかし,オブジェクトの設定でキャッシュを利用するかコントロール可能なため,キャッシュを利用しないように指定している.なお,実験を中止したい場合は,強制終了ボタンを押すことで,実験の測定を中止し初期画面である実験パラメータ入力画面へ遷移する.バッテリ残量情報については,実験画面でのみ表示,CPU 使用時間などの Workload 情報は,どの画面でも表示される.

4.2.2 モデル機能

モデルは以下の実装を行った.

- 実験パラメータを NSUserDefaults[9]で Key-Value 形式管理
- 実験対象を測定し測定値を管理する機能
- 実験結果を CSV ファイルに出力する機能

ビューで入力された実験パラメータを保存し,どの画面からでもアクセス可能である必要がある.そこで,バッテリ残量を取得するメソッドと測定値を保存するメソッドを実装した.NSUserDefaults を使用した.実験パラメータと実験結果が保存されるファイルの実態は plist ファイルであり,アプリケーションが終了してもデータが保持可能である.バッテリ残量を取得するメソッドを実行すると,測定値を保存するメソッドも実行されるため,バッテリ残量を取得するメソッドは任意のタイミングで遅延実行している.バッテリ残量を取得するメソッドでは,バッテリ残量を取得する API を 2 つ使用している.1 つは公式 API であり,5%区切りでバッテリ残量を取得可能.もう 1 つは非公式 API であり,±3%の誤差でバッテリ残量を取得可能.以上の 2 つで測定を行っている.その後に,測定値を保存するメソッドが呼ばれる.このメソッドでは,NSFileManager[10]を利用して,CSV ファイルの出力を行っている.流れとしては,CSV ファイルとして書き込むデータの準備を行い,CSV ファイルとして書き込みを行っている.

4.2.3 コントローラ機能

コントローラは以下の実装を行った.

- TestStart(UIButton)が押されたときに実験開始要求
- Web ブラウザ操作要求
- バッテリ残量が設定値以下になった時の実験終了要求
- 強制終了(UIButton)が押されたときに実験強制終了要求
- 実験パラメータ管理要求

実験パラメータ入力画面で TestStart が押された通知を受け取ると,実験パラメータの保存,実験パラメータの読み出し,実験画面へ遷移,測定の開始といった命令を実行する.実

験画面はほぼ Web ブラウザなので,Web ページを操作することがある.そこで,Web ブラウザの機能である「進む」「戻る」「更新」「中止」を行いたい場合に通知を行えるように実装した.各通知を読み取ると,それぞれ機能にあった処理が実行される.画面操作で実験を強制終了するか,バッテリ残量が設定した閾値以下になると通知が行われる.通知を受け取ると,実験終了する処理が行われる.実験を終了する処理とは,測定メソッドの遅延実行停止,Web ページを何もないブランクなページへ移動,測定データを CSV ファイルへ出力停止,実験パラメータ入力画面へ遷移である.実験画面に右下へ表示されるボタンである.実験パラメータを間違えて設定した場合や対象 Web ページが表示されないなどのトラブルに備えて実装した機能である.強制終了ボタンを押すと,実験終了通知が行われる.そのため,機能は同じだが,測定データは消去される仕様になっている.実験パラメータを保存したり読み出したりする場合に実験パラメータ管理要求が行われる.受け取ると,NSUserDefaults での管理が行われる.実験パラメータを読み出して実験画面の設定をするように通知を行う.また,測定の開始要求,ファイル出力開始通知を行う.測定終了通知,ファイル出力終了通知を行う.実験パラメータ初期化通知,ファイル消去通知を行う.

4.3 Android

4.3.1 ビュー

Android 版では画面輝度と測定対象の URL のみ設定な実験パラメータ入力画面が実装されている.初期画面は iOS のものと同じである.次に,実験画面は,WebView を実装したが,「戻る」「進む」「更新」「中止」といった機能は未実装のため使用することができない.理由としては,Android 本体に存在する戻るボタンで実験パラメータ入力画面へ戻ることで実験を中止が可能なためである.そのため,強制終了ボタンの実装も行っていない.ページの読み込みに失敗した場合は,実験を中止して再度実験を開始する必要がある.

4.3.2 モデル

Android では,iOS の NSUserDefaults と同じような API が存在し,SharedPreferences と呼ばれ,iOS と同じく Key-Value 形式でデータの管理が可能となる[11].バッテリ残量は,公式 API で正確な値を取得する事が可能であったため,公式 API のみの値である.測定のタイミングは設定した値ではなく,OS から任意のタイミングで実行されるため,定期的な実行ではない.しかし,操作を行わなければ,ある程度定期的に呼ばれているため,測定には影響がないものとして考えている.だが,iOS と同様とするためにも測定メソッドの遅延実行を一定時間間隔で動作させた方がいいので,修正する予定である.CPU 使用率については,Linux と同じようにシステムのファイルから値を取り出して求めている.

4.3.3 コントローラ

画面の操作や,バッテリ残量の値などによって実験開始,実験終了,測定開始,測定終了,ファイル出力開始,ファイル出力終了,Web ページ読み込み,といった要求があったことを通知する.実験開始通知を受け取ると,設定した実験パラメータを保存する.保存された実験パラメータを元に実験画面へ遷移し,実験を開始する.実験終了通知通知を受け取ると,実験画面の破棄を行うと同時に,測定終了通知を行う.測定終了通知を受け取ると,バッテリ状態の変化によって発生していたメソッドを発生しないようにする.ファイル出力開始通知を受け取ると,実験結果データを CSV ファイルとして出力を開始する.ファイル出力終了通知を受け取ると,CSV ファイルの出力を停止する.Web ページの読み込み要求では,設定した URL を読み込む.なお,ページ上移動は可能である.

5. 予備実験

5.1 予備実験方法

予備実験は,2 つのパラメータで実施した.1 つ目は画面輝度で,端末自体の画面の明るさがどの程度影響するのか調べる.2 つ目は fps でフレームレートが変化すると,アプリケーションが受信するデータ量にも影響がでるため,この電力消費量との関係性を調査する.

表 7,表 8 に予備実験パラメータを示す.これらのパラメータにより電力消費量の測定を実施し,端末本体に保存された実験結果をグラフ化した.

Web ページ上でのオンラインビデオ再生を実験対象とした.このビデオデータは,表 6 に示す条件で研究室から撮影したものである.そのビデオデータは HTML5 で作成されたオンラインビデオ再生ページと一緒に,研究室のサーバへ設置した.

表 6 ビデオデータの詳細

撮影機材		Canon iVIS HFM51
撮影条件	場所	研究室
	撮影時間	12時間
	音声	なし
	フレームレート	60fps

表 7 iOS での予備実験パラメータ

	フレームレート[fps]	画面輝度[%]
実験1	30	100
実験2	12	100
実験3	30	50
実験4	12	50

表 8　Android での予備実験パラメータ

	フレームレート[fps]	画面輝度[%]
実験5	30	100
実験6	12	100
実験7	30	50
実験8	12	50
実験9	30	0
実験10	12	0

5.2 予備実験環境

本論文で実施した予備実験の環境を図 2 に示す.また,予備実験で使用した端末のスペックを
表 9 に示す.Macmini は Web サーバとなっており,研究室内 LAN,TimeCapsule までは有線 LAN でリンクとなるため速度は 1Gbps である.TimeCapsule とクライアント間は無線 LAN となっており,54Mbps となっている.実験で使用した端末に関しては,
表 9 の通りである.

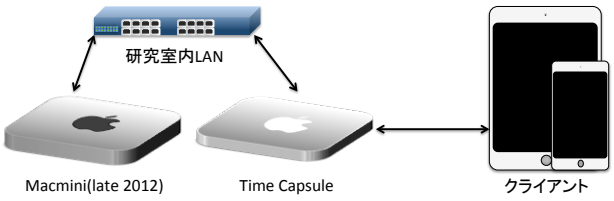

図 2　実験環境図

表 9　実験で使用した端末詳細

構成要素	iPadmini(第1世代 2012年モデル)	Nexus7 (第2世代 2013年モデル)	Macmini(2013年モデル)
CPU	AppleA5 2Core 1.0Ghz	Krait 1.50GHz (Qualcomm Snapdragon S4 Proに内蔵])	Intel Core i7 2.6GHz
GPU	PowerVR SGX 543MP2 2Core	Adreno 320 400MHz (Snapdragon S4 Proに内蔵)	Intel HD Graphics 4000
RAM	512MB(LPDDR2)	2GB(DDR3L)	16GB (DDR3)
NIC	Wi-Fi (802.11 a/b/g/n)	Wi-Fi (802.11 a/b/g/n 2.4 GHz & 5 GHz)	10/100/1000BASE-Tギガビットethernet Wi-Fi()
Power	電圧3.75ボルト 16.3ワット時のリチウムポリマー電池 最長10時間のバッテリー駆動 4400mAh	交換不可・内蔵充電式リチウムイオンポリマーバッテリー (3950 mAh, 16 Wh) 無線給電 "Qi"	AC 100V
Screen	7.9インチ型TFT(IPS)マルチタッチ液晶 XGA (eXtended Graphics Array) 画素数:1024x768ピクセル 解像度:163ppi	7.02インチ (178 mm) 静電容量方式タッチパネル付きIPS液晶ディスプレイ アスペクト比16:10ワイドスクリーン 1920×1200ピクセル (323 ppi) コーニング製 耐キズガラスパネル	-

5.3 予備実験結果

図 3 に iOS 版での予備実験結果を,図 4 に Android 版での予備実験結果を示す.

iOS では,図 3 に示す通り,実験 1 と実験 2 のグラフも,実験 3 と実験 4 のグラフもほぼ同じで結果となっている.実験 1 と実験 2 では画面輝度の条件が同じ 100%であり,異なる点はフレームレートである.実験 3 と実験 4 についても同様に,画面輝度が 50%であり,フレームレートが異なる.さらに,実験 1-4 の CPU 使用率を確認したところ,最大でも 10%程度の利用率であるため,CPU をほとんど使用していない事がわかる.そのため,iOS 端末では,デコードの影響がほとんどなく,動画を表示する画面(液晶)による電力消費量が多いことがわかる.

Android 端末では,図 4 に示す通り,iOS 端末とは異なる結果が得られている.フレームレートが 30[fps]の時では,最大 5 時間,最小 3 時間再生可能であった.再生時間が最大となる条件は画面輝度が 0%であり,最小となる条件は画面輝度が 100%である.画面輝度が 50%の時では,最大と最小時間の 50％である 4 時間を示している.次にフレームレートが 12[fps]の時では,30[fps]の時と同様に,画面輝度が 100%の時は 8 時間再生,画面輝度が 0%の時は 4 時間再生となっており,画面輝度が 50%の時は,やはり再生時間の最大と最小の約半分である,5.5 時間である.現段階では,CPU 使用率が Android 版で取得できていないためデコードの CPU 使用率への影響について明らかとなっていない.また,画面輝度は iOS 端末と同様に発生し,0%-100%では 2 倍の差があることがわかった.

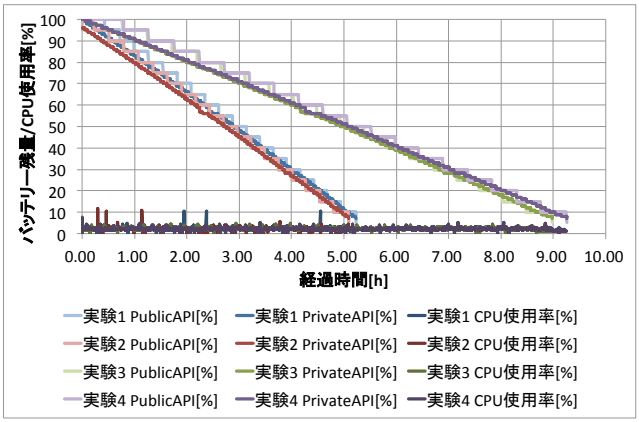

図 3　iOS 版での予備実験結果

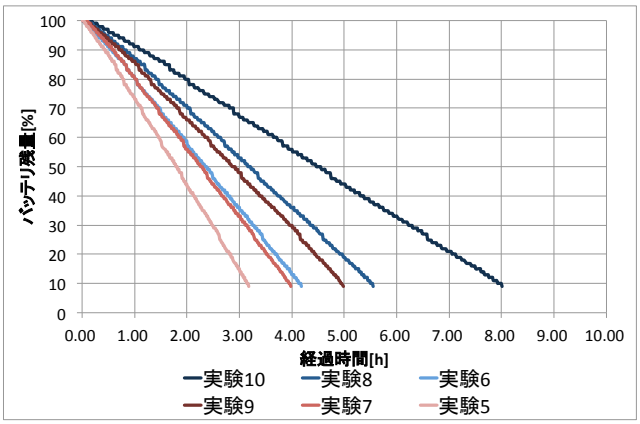

図 4　Android 版での予備実験結果

6. まとめ

予備実験全体の結果としては,スマートフォンとして主流である iOS 端末と Android 端末での電力消費量測定用 Web ブラウザの開発ができ,Web ページ閲覧時の電力消費量測定が可能となった.

6.1 フレームレートによるデコードの影響

iOS 端末では,図 3 よりフレームレートによる再生時間の変化はほとんどなく,CPU 使用率がどの実験でも 10%程度のため,CPU が動画のデコードに関する処理を実行して

ないことがわかる.一方,Android 端末では,図 4 よりフレームレートによって再生時間が変化する結果となった.そのため,フレームレートによる電力消費量への影響が発生している.

6.2 画面輝度による影響

液晶画面はスマートフォンの部品の中でも重要であり,利用者に視覚的にデータを表現することが可能である.液晶画面では,バックライトが点灯することによって画面が明るくなり,色を表現する部分で着色を行っている.そのため,画面輝度は,バックライトの明るさで変化させている.明るければ電力消費量が増えることは予備実験で明らかにした通りである.このため,画面輝度の制御を,電力消費を抑えるように制御できれば,電力消費量削減につながると考えられる.

6.3 今後の課題

今後の課題としては,本実験の検討と実施,異なるプラットフォームでの実験,1 対 1 でビデオチャット可能なアプリケーションの開発とそれを使用した電力消費量の測定が挙げられる.

本実験の検討と実施では,予備実験の結果を踏まえて実験パラメータの再検討と,開発したアプリケーションの改良,実験環境の再検討,測定対象の再検討が挙げられる.理由としては,コミュニケーションサービス全体の電力消費量を削減する事が本研究の目的である.そのため,iOS,Android 端末毎に電力消費量が大きい処理を特定するには,電力消費量が大きい処理が特定できる実験方法と実験パラメータ,電力消費量測定アプリケーションの改良が必要である.

今後,本実験に向けて機能の追加やバグフィックス,ブラッシュアップを行う予定である.さらに,1 対 1 でビデオチャット可能なアプリケーションの開発とそれを使用した電力消費量の測定も今後実施していきたい.

参考文献

1) A, Hooper, ”Green computing,” Communication of the ACM, vol.51(10),pp. 11-13,2008.
2) 白鳥則郎, 稲葉勉, 中村直毅, 菅沼拓夫, "災害に強いグリーン指向ネバーダイ・ネットワーク", 情報処理学会論文誌, Vol.53, No.7, 1821-1831, July 2012
3) 古市 実裕,相原 達:アプリケーションプログラム参加の省電力制御に必要な機能の考察, IPSJ OS/DPS 合同研究会, Vol.98, No.15, pp.135-140, (1998)
4) 藤本恭平, 安本慶一, 孫為華, 山内由紀子, 伊藤実, “オブジェクトの監視・追跡を行う無線マルチメディアセンサネットワークの稼働時間延長および QoS 確保のためのルーティング手法,” DICOMO2011 シンポジウム論文集, pp1156-1166, 2011.7.6.
5) 山岡修一, 孫タオ, 玉井森彦, 柴田直樹, 伊藤実, “多様な要求品質を持つ移動端末ユーザへのリソース効率の良いビデオ配信方式(QoS)”, 情報処理学会研究報告. MBL, [モバイルコンピューティングとユビキタス通信研究会研究報告] 2005(113), 77-84, 2005-11-17
6) 安本 慶一, 小倉 和也, 山本 眞也, 伊藤 実, ”快適度の低下を最小限に抑える省エネデバイス制御手法”, 第 149 回 マルチメディア通信と分散処理(DPS)研究会, 情報処理学会研究報告 Vol.2011-DPS-149 No.9, pp. 1-8, (November 2011).
7) HTTP Live Streaming の概要 https://developer.apple.com/jp/devcenter/ios/library/decumentation/StreamingMediaGuide.pdf (2013)
8) View Controller Programming Guide for iOS https://developer.apple.com/library/ios/featuredarticles/ViewControllerPGforiPhoneOS/ViewControllerPGforiOS.pdf (2014)
9) Preferences and Settings Programming Guide https://developer.apple.com/library/ios/documentation/Cocoa/Conceptual/UserDefaults/UserDefaults.pdf (2014)
10) File System Programming Guide https://developer.apple.com/library/ios/documentation/FileManagement/Conceptual/FileSystemProgrammingGuide/FileSystemProgrammingGuide.pdf (2014)
11) Public interface SharedPreferences http://developer.android.com/reference/android/content/SharedPreferences.html (2014)