PC環境での心理学実験におけるレイテンシとジッタの再検証

Revalidation of Latency and Jitter in Psychological Experiments in a PC Environment

長嶋　洋一

YOICHI NAGASHIMA

静岡文化芸術大学デザイン学部

Shizuoka University of Art and Culture

nagasm@suac.ac.jp

内容梗概：約20年前に多くの機器/システムの遅延(レイテンシ)とばらつき(ジッタ)について計測実験を行った。そして、音楽心理学実験の領域で便利/高性能な「道具」として普及したテクノロジーを正しく理解せずに使用することは、研究の基盤そのものを無意味にしてしまうと警鐘を鳴らした。現在では当時に比べてPCの性能が約100倍ほど向上し、多種の周辺環境が整備されてきたが、情報技術/情報科学の本質としてのレイテンシ/ジッタは依然として存在している。そこでネットワーク環境や物理コンピューティングの進展を受けて、改めて当時と同様の問題点に関する計測実験を行ってみたので報告する。

Key Words : Latency, Jitter, Max/MSP, Arduino, Psychological Experiments

1. はじめに

心理学実験を行なうPCベースでのマルチメディア・システムの遅延(レイテンシ)とばらつき(ジッタ)について改めて計測実験を行い、研究の基盤に関わるポイントについて再検討したので、本稿では主として音源の発音遅延の検証実験について報告する。OSC伝送の検証実験の詳細は[1]を参照されたい。また、約20年前に行なった実験と検討(心理学実験への警鐘)については[2-6]を参照されたい。

まず最初に用語を確認しておくと、レイテンシとは、デバイス/システムに対してデータ転送などを要求してから、その結果が返送されるまでの不顕性の高い、原理的な理由により発生してしまう本質的な遅延時間のことである。一般的にレイテンシが小さければ小さいほど、そのデバイス/システムは高性能で高価である。

ジッタとは、電気通信などの分野において、時間軸方向での信号波形の揺らぎの事であり、その揺らぎによって生じる映像等の乱れのことも指す。デジタル信号でのジッタは、ランダムジッタ(正規分布に従う時間軸方向での信号波形のランダムな時間的揺らぎ)とデターミニスティックジッタ(データやクロックに依存して受信信号の波形タイミングが本質的に変化するジッタ)に分類される。

今回の実験を思い立った動機は、ふとPCのターミナル(コンソール)で「ps -A」と入力してみた事がきっかけである[1]。現代のPCはWindowsであれMacであれLinuxであれ、全て基本的にはUnixベースのマルチタスクOSであり、使用/起動しているアプリケーションとは別に、システムに関する多数のプロセスがバックグラウンドで走っていて時分割多重化処理(不要になったメモリ空間を掃除するガベージコレクション[その間は他処理が停止する]を含む)を行っている。通常時に同時に200本以上のプロセスが常に裏で走っていると知って、このような環境でのレイテンシとジッタが気になって調べてみる事になった。

2. 実験の概要

2.1. 実験環境

実験に使用した5台のコンピュータは全てMacOS10.11.6(El Capitan)のMacである。実験を行うメインMac mini (2.6GHz Intel Core i5, 8GB 1600MHz DDR3)およびOSCメッセージを返すサブiMac (Retina 5K, 3.2GHz Intel Core i5, 16GB 1867MHz DDR3)、3台のMacBookAir (1.4GHz Intel Core i5, 4GB 1600MHz DDR3) (2.2GHz Intel Core i7, 8GB 1600MHz DDR3) (1.3GHz Intel Core i5, 8GB 1600MHz DDR3) を使用した。インストールされているMaxはMax7.3.6とMax8.0.2であり、時間的性能は最高速度(Overdrive, scheduler=1msec, slot=1msec)に設定した。メインMac miniにUSB接続した周辺機器

は2種であり、「115200bpsのUSBシリアルでMaxから受け取った4種類のデータに応じて2ビットディジタル出力ポートを上げ下げするだけ」のスケッチを書き込んだArduino UNO、そしてUSB接続された2チャンネル100MHzディジタルサンプリングストレージオシロスコープ(英国Pico Technology社製PicoScope3205A)である。実験は全てMax環境において実施し、スタートのタイミングでまずArduino UNOの1ビットをHからLに変化させ、この信号をPicoScope3205Aは第1チャンネルのワンショット・トリガモード(立ち下がりエッジ)でサンプリング開始して、第2チャンネルがどう変化したかの記録を遅延イベントの時間として計測した。例えばMax側でスタートから2msecの遅延(delayオブジェクト)で第2チャンネルを変化させて、トリガからの時間差を10数回の試行で遅延(レイテンシ)のばらつき(ジッタ)を計測し、正確な2msecの遅延とジッタがほぼゼロであることから「道具」として使えることを確認した。Fig.1はその装置の様子、Fig.2は実験1の画面の一例である。

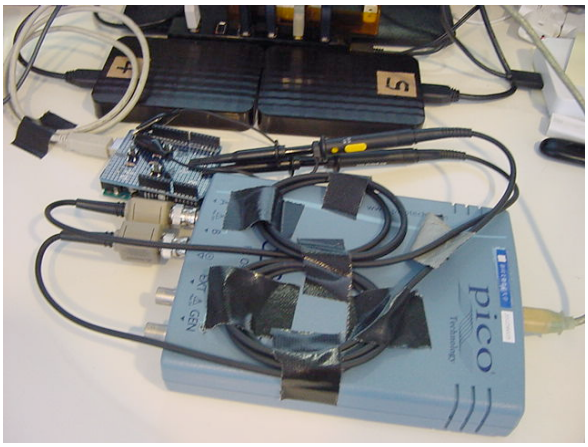

Fig. 1 Arduino UNOとPicoScope3205A

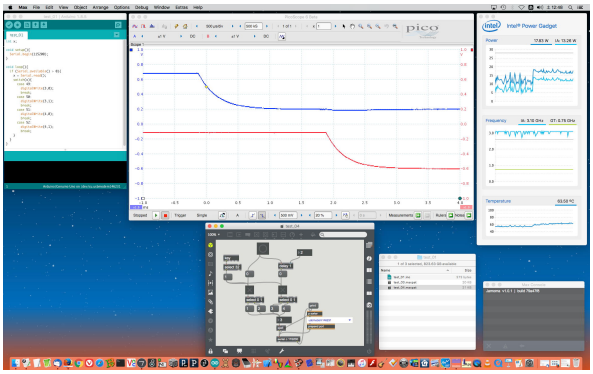

Fig. 2 実験1の画面の一例(2msec)

2.2. OSCの遅延実験

上述の実験環境の確認に続いて今回行った検証実験は全部で8種類あり、OSCに関する実験とMaxから鳴らす音源の発音遅延に関する実験とに大別される。このうちOSCに関する実験1～実験5については、情報処理学会音楽情報科学研究会・音学シンポジウム(2019年6月)にて詳細に報告する予定なので[1]、本稿ではここでごく簡単に紹介しておくにとどめる。

Open Sound Control (OSC)とは、スタンフォード大のCNMATが提唱したネットワークプロトコルであり、現在ではMaxやFlashやProcessingやUnityやPythonやOpenCVやRaspberry Piなどマルチプラットフォーム・プロトコルとして世界標準になっている。UDPパケットとして高速ネットワークを活用して多種のシステムが連携できるので、心理学実験として多くのシステムが共存して高度なシステムを構築する道具としてのOSCのレイテンシとジッタについて調べておきたい、というのが目標である。

最初の基本的な実験1として、静岡文化芸術大学(SUAC)北棟11階の筆者の1106研究室のMac(Mac mini)からSUAC学内ネットワーク(LAN)を介してSUAC南棟4階のマルチメディア室のMac(iMac)との間(経路100メートル以上)でOSCメッセージ"0"を往復させてレイテンシを計測し、複数回の試行によってジッタの傾向を調べた。その結果、「OSCで1往復」の実験のレイテンシは約1.0msecであり、複数回の試行でばらつき(ジッタ)が無いことを確認した。さらに「OSCで10往復」の実験のレイテンシは10～12msecあたり、「OSCで100往復」の実験のレイテンシは118～125msecあたりとなり、OSCが高速インターフェースとして活用できることをまず確認した。さらに1バイトの"0"でなく「592文字/107ワードの英文」というOSCパケットでの実験3により、メッセージが多量になっても性能は変化しないことを確認した。

OSCに関する他の実験では、OSCメッセージが行き来するネットワークのトラフィックを過密にした時の影響、さらに実験を行っているMacが多数の「重い」処理を同時並行した場合の影響について調べたが、ここではもっとも顕著な結論だけ述べると、バックグラウンド(同時並行)にて16本の大容量データをインターネットからダウンロードする(ネットワーク負荷+コンピュータ内部負荷)という実験2では、OSC伝送のレイテンシが大幅に増加するだけでなく、そのばらつき(ジッタ)が非常に大きくなり、音楽情報科学的に問題となる数十msec以上になることを確認した。

2.3. Maxで10ミリ秒の純音(1000Hz)バーストを出す実験

約20年前に行なった実験と検討[2-6]においては、音源としてMIDI電子楽器が主流の時代だったので、多くのメーカの多くの製品につ

いて発音遅延とそのばらつきを計測した。しかし現在ではPCの性能向上によって、PC内部で電子音でも楽器音でも豊富にリアルタイム生成できるようになったので、今回の実験では「MIDI経由で音源モジュールを鳴らす」というような古風な手法は捨象した。この実験6では、Max/MSPとして実装されているリアルタイムサウンド合成の出力信号を対象として、PicoScope3205Aの第2チャンネルに Mac miniのオーディオ出力を接続してサウンドの立ち上がりまでの遅延(レイテンシ)を計測し、そのばらつき(ジッタ)を調べた。つまりFig.3のように、Arduinoの「スタート」タイミングと同時に”cycle~”オブジェクトで1000Hzの純音をONしてから10msec後にOFFして、スタートからの発音遅延とともにMax7内で10msecとしているON→OFFの幅(duration)も調べた。すると、イベント遅延については「17msec～25msec」というような幅があり、10msecバースト音の幅(duration)については「11msec～20msec」というような幅があった。Max7内のMSPで、bufferメモリにデータを詰めてからCoreAudioに送るため、発音イベントの遅延量としてはまずまず妥当とも思えるが、ばらつき(ジッタ)もかなりあった。さらにMax7内で「10msec固定」の筈のバースト音の幅が、約2倍というような幅でのばらつき(ジッタ)があったのは予想外であった。

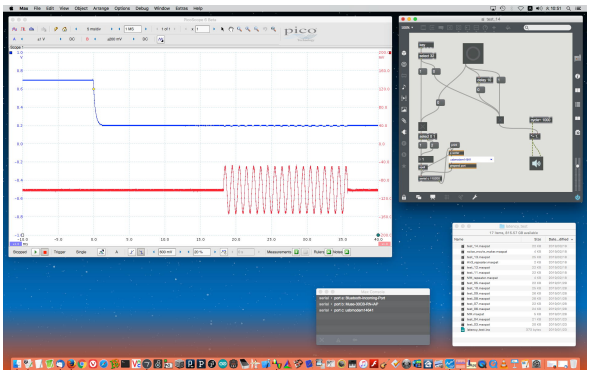

Fig. 3 実験6の画面の一例(Max7)

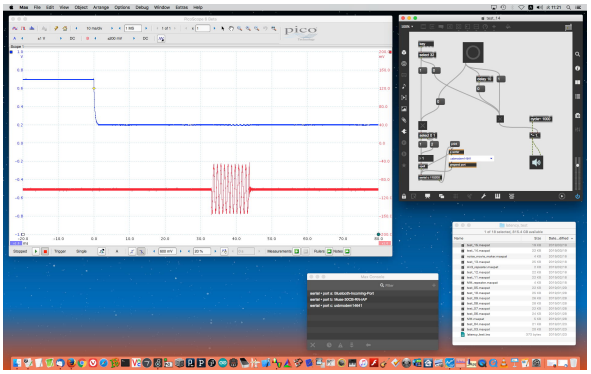

Fig. 4 実験6の画面の一例(Max8)

ここでMax7に代えて、2018年9月に発表された新しいMax8を使って、まったく同じ実験を行ってみた。するとFig.4のように、Max8ではイベントの遅延については「24msec～33msec」というような幅があってMax7よりもさらに遅かったのだが、何度試行してもバースト音の幅がほぼ「きっちり10msec」となった。おそらくMax7からMax8に改良される中で、「MSPでbufferメモリにデータを詰めてからCoreAudioに送る遅延」というあたりの正確性のpriorityを上げていると推察できた。

2.4. Max7から内蔵音源(DLSシンセ)のピアノ音を鳴らす実験

次に行った実験7では、MSPのサウンドでなく、Max7からMIDI対応の内蔵音源(DLSシンセ)のピアノ音を鳴らすという実験を行った。Maxパッチとしては、MSPの”cycle~”の代わりに、”noteon 127 30”という「最大ベロシティの短い音」で「ノートナンバ=96」を鳴らすものとした。このDLSシンセ音源は全体として音量が小さいので、トラックボリュームも最大の127とした。Fig.5はその実験画面の一例であるが、Max7ではイベントの遅延については「15msec～23msec」というような幅があり、”cycle~”よりもむしろ早かった(遅れが小さかった)。これは内部的にDLSシンセにトリガ送出するだけ(→あとはDLSシンセが裏で何かやっている)というためと思われ、ばらつき(ジッタ)はだいぶ小さく良好であった。

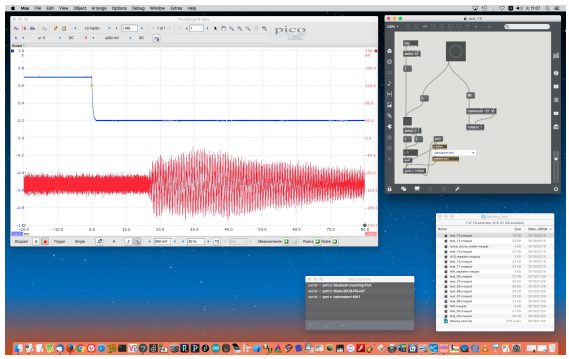

Fig. 5 実験7の画面の一例

次に同様に、Max7でなくMax8から内蔵音源(DLSシンセ)のピアノ音を鳴らすという実験を行った。Max8ではイベントの遅延については「15msec～19msec」というような幅で、遅延のレンジとしてはほぼMax7と同等であったが、何度やっても20msecまで遅れることがなく、発音遅延のジッタ特性という意味では確かに改良されていた。

2.5. Max7からGarageBandを内部MIDI音源として鳴らす実験

最近ではインスタレーション作品などのメディアアートのサウンド要素の生成に、Macに標準となっているGarageBandを使うことが増

えている。MaxからもDLSシンセだけでなく、GarageBandをMIDI音源として内部的に鳴らすことが出来る。Max起動中にGarageBandを起動すると、「いまMIDI入力機器として〇台のデバイスを確認」などと表示されるが、これはMacOSとMaxとが内部的に連携している証拠である。環境設定としてMaxからMIDI送りする行先のオブジェクトとして「IAC Driver(内部)バス」を選ぶだけで、同時に起動しているGarageBandを、defaultのDLSシンセとは別の内部音源として利用できる。そこで実験8として、実験7の内蔵音源(DLSシンセ)からGarageBandのピアノ音を鳴らす場合の発音遅延とジッタを調べた。

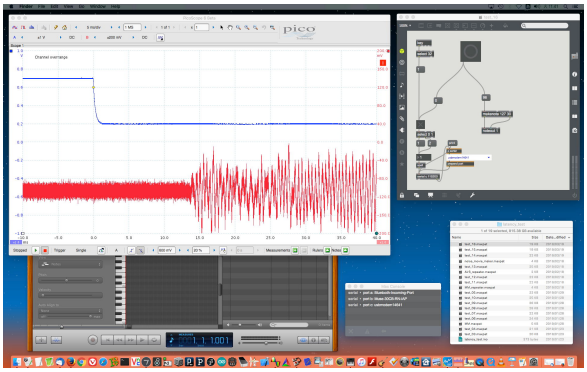

Fig. 6 実験8の画面の一例

Fig.6はその実験画面の一例であり、Max7では発音のレイテンシ13msecという短時間で、GarageBandが(別プロセスとして)defaultのリバーブまでかけたピアノ音を鳴らした。Max7実験でのイベントの遅延は「12msec～18msec」というような幅で、内蔵音源(DLSシンセ)のピアノ音よりも成績はまずまず良好だった。同一の条件でMax7からMax8にしてMax7とほぼ同等の結果となった。リバーブやイコライジングなどの豊富なエフェクトまで(OFFとしても)付加されてしまうGarageBand音源であるが、レイテンシ/ジッタの面で、実験用音源としては十分に使えると確認できた。

3. 結論と考察

音楽心理学実験に関するPC環境での計測実験ツールとして「MaxとArduinoとサンプリングオシロスコープPicoScope3205A」という組み合わせは十分な性能がある(「遅延時間計測システム」として1msec精度の計測が可能)とまず確認できた。

マルチプラットフォーム・プロトコルとして世界標準になったOSC(OpenSoundControl)によるメッセージ伝送は、SUACネットワーク環境において、OSCで1往復のレイテンシは約1.0msecであり、複数回の試行でばらつき(ジッタ)が無いことを確認した。

Max/MSPのリアルタイム音響処理機能による純音バースト(10msec)生成実験から、Max7では発音遅延およびバースト幅のレイテンシおよびジッタの傾向に、内部タイミング処理の影響からか問題があると判断した。Max8では発音遅延はさらに大きくなったものの、バースト幅はほぼジッタ無しの正確なdurationを確保するよう改善されていた。これらレイテンシのオーダ(絶対値)は過去の音楽心理学実験において「人間が音色の違いと認識する」レベル(msecオーダ以上)なので、Max内で設定したつもりの時間条件で音楽心理学実験を行うことが無意味となる危険性がとても高い。

MaxからMIDI機能でMac内蔵音源(DLSシンセ)およびGarageBandのピアノ音源のピアノ音を鳴らす実験から、音源処理をMacOSに引き渡すことで、ジッタの幅はあるものの発音遅延はMax7/Max8ともにより良好になった。しかし音楽心理学実験として問題あるオーダでのジッタは存在しているので注意が必要である

4. おわりに

心理学実験を行なうPCベースでのマルチメディア・システムの遅延(レイテンシ)とばらつき(ジッタ)について計測実験を行い、研究の基盤に関わるポイントについて再検討(本稿では主として音源の発音遅延の検証実験について報告)した。20年前と同じ結論であるが、お手軽な道具としてのPC環境の利用には十分な注意と検討が必要であろう。

参考文献

[1] http://nagasm.org/ASL/Latency_Jitter/

[2] 長嶋洋一, ハード音源/ソフト音源のMIDI発音遅延と音楽心理学実験環境における問題点の検討, 平成11年度前期全国大会講演論文集2, 情報処理学会, 1999.

[3] 長嶋洋一, MIDI音源の発音遅延と音源アルゴリズムに関する検討, 情報処理学会研究報告 Vol.99,No.68 (99-MUS-31), 情報処理学会, 1999.

[4] 長嶋洋一, MIDI音源の発音遅延と音楽心理学実験への影響, 日本音響学会音楽音響研究会資料 Vol.18,No.5, 日本音響学会, 1999.

[5] 長嶋洋一, 生体センサによるパフォーマンスとシステムの遅延/レスポンスについて, 平成14年度前期全国大会講演論文集4, 情報処理学会, 2002.

[6] Yoichi Nagashima, Measurement of Latency in Interactive Multimedia Art, Proceedings of International Conference on New Interfaces for Musical Expression, NIME, 2004.

[7] http://opensoundcontrol.org/