報告

未踏の第 26 期 スーパークリエータたち

竹内郁雄 | IPA 未踏 IT 人材発掘・育成事業　統括プロジェクトマネージャ

未踏事業で採択され，優れた成果や成長を示した人たちを未踏スーパークリエータと呼ぶ．この認定は 2020 年で 26 回目となる．突出した才能を持つスーパークリエータを広く産業界や学界に知っていただきたい，というのがこの年次報告の狙いである．

第 26 期の未踏クリエータは計 26 名（20 プロジェクト）で，そのうちの 17 名（13 プロジェクト）がスーパークリエータとして認定された．2018 年の認定率 59.3％をさらに超えた 65.4％となった．喜ばしい．今期は高校生 1 名，女性 1 名が認定された．

今期は低レイヤから Web アプリまで万遍なく分布したバランスの良い配分となった．

クリエータを代表者の 50 音順に紹介する．タイトルは正式なものではなく，「名は体を表す」キャッチに変えてある．なお，2020 年 2 月 15 ～ 16 日の 2 日間に開催された成果報告会（Demo Day）のすべての動画は https://www.youtube.com/user/ipajp/ から見ることができる．最近のプロジェクトはデモなど，動画で見ないと面白さや意義が分からないものが多いので，この記事は単なる導入紹介とご理解いただきたい．またプロジェクト件数が多いので，例年より短い記述になることをお許し願いたい．

市川 友貴（いちかわ ゆうき）

イチゴの自動受粉ロボット☆1

従来，イチゴ栽培における受粉作業はミツバチによって行われてきた．このミツバチは輸入に頼っており，最近は大量死などがあり価格が急上昇している．日本ミツバチは逃亡癖があり使えないらしい．近年工場栽培が盛んになっているが，工場内ではミツバチを飛ばすことができないため，果物工場はほとんど存在しない．

市川君はイチゴ農園で，ミツバチに代わって，ロボットに受粉させるロボットの開発に挑んだ．レールの上を走るロボットの深度カメラによってイチゴの花の色・深度情報を観測し，適切な受粉時期を判断し，アームの先端についたブラシで受粉を行う．

概要だけを書くと，えらく簡単そうに見えるが．相手はやわい生き物である．想像以上の苦労があった．写真 1 の HarvestX という装置はイチゴの水耕栽培を行う．水平の塩ビパイプにイチゴを植え付け，パイプ内に液肥を希釈させた水を 24 時間流すことでイチゴに水分の供給を行う．赤紫色の LED は毎日 9 時から 17 時に点灯するようにした．大変だったのは温度管理で，サーバルームに保管して温度が上がりすぎないようにし，1 週間に 2，3 回，20 度程度のスポット送風もした．これは，花芽分化を促進させるためである．

花の認識は，花の開く方向が同じになるように仕向けてあるので，ある程度楽なのだが，受粉用ブラ

☆1 https://harvestx.jp/ja/

シを大きな誤差なく花の中心（雄しべと雌しべ）に持っていくのは，認識とロボットアーム制御の密接な連動を実現しないといけないので，結構面倒なプログラミングが必要である．

こうして，成果報告会までに世界初のロボット受粉によるイチゴを少数だが実らせることができた（写真 2）．食べた人の感想によると，甘みが強くてとても美味しかったそうである．

ところで，イチゴの受粉は花粉を雌しべに単にくっつけるだけではだめで，均等にくっつけることが必要である．均等でないといびつな形のイチゴになってしまう．逆にこのことを利用すれば，ハート型や四角いカスタムイチゴを生産できることを意味する．これは今後の課題だ．

ブラシの材質もいろいろ検討しなければならない．耳掻きの尻尾についている水鳥の羽を使った梵天が良さそうだということが分かっている（ミツバチの体毛に近いらしい）．このほかにまだまだやるべきことが多いが，市川君は持ち前の行動力で多数のイチゴ農家の協力を取り付けている．

この装置が実用化されれば，一定間隔でイチゴを実らせて味を保証できる．また 24 時間体制で受粉ができ，ミツバチでは不可能な収穫量・出荷量の制御が可能になる．また植物工場の特徴である無菌栽培も可能になる．まさに良いことずくめだ．

市川君は未踏終了後，米国のファームで武者修行をする予定だったが，新型コロナ騒ぎのため予定を変更して，国内での起業を前倒しすることにした．市川ロボットイチゴの出荷が楽しみだ．

（五十嵐悠紀 PM（プロジェクトマネージャ）担当）

■上田 裕己（うえだ ゆうき）

ソースコード修正作業を代行するボット☆2

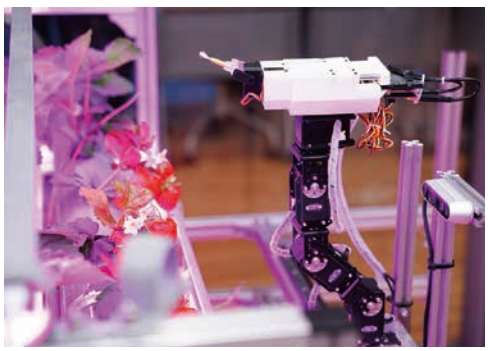
ソフトウェアの信頼性を確保することは昔から難しい作業である．最近はコンパイラのエラーチェックが随分進化してきたが，エラーにならなくても，開発グループの中で「書き方のスタイル」を統一しないと，保守性が下がる．構文エラーにならないプログラムをチェックしてくれる代表格は C 言語の Lint だが，上田君が開発した DevReplay は，名前の通り，開発者が意図を明言して行ったソースコードの修正を記録し，類似のものがコードの中にあったら自動的に同じ修正を適用してくれるシステムである．

典型的なデバッグはもちろん，スタイルを統一するのにも DevReplay は役に立つ．さらに実行速度の優れたソフトウェアの開発履歴を利用して，自分のコードの性能を上げることも可能になる．

Python のコード修正の典型的な例を図 -1 に示す．左が修正前のコードで，右には対応するルールが表示されている．condition はパターンマッチの条件で，consequent が修正後のパターン，description（修正方法の詳細コメント），severity（重要度），author などが続く．

具体的には，7 行目が if 文の曖昧な書き方の修正である．勝手に修正するわけではなく，画面上に電

☆2 http://devreplay.github.io/



■写真 1
水耕栽培装置 HarvestX のロボット

■写真 2
世界で初めてロボットが受粉したイチゴ

球のようなアイコンが出て，それをクリックすると修正される．また 3 行目以下は値のスワップの書き方が気持ち良くなる修正である．

似たことをしてくれるツールに，Lint, IntelliCode, Sleek, APR ツールなどが存在するが，DevReplay の特徴は，プロジェクトの暗黙的なソースコード改善パターンを新たに発見する点にある．ソフトウェアを構成するソースコードの記述内容は，そのときの採用技術や開発チームの方針によって変化するが，DevReplay はソースコードの修正履歴から，固有名詞や社内ルールのような開発チームの中で閉じた修正も自動的に実施可能にする．

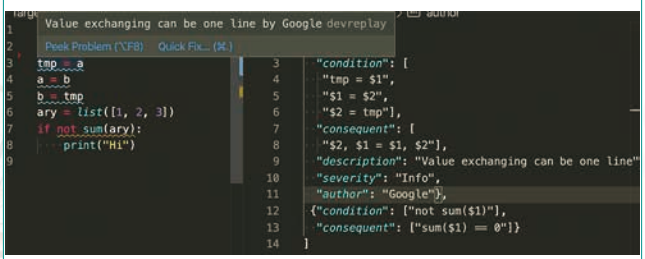
ただし，すべてを自動化するわけではなく，開発者が上で述べたような修正ルールを編集することも可能である．システムが自動的に発見したルールをしっかり補強できるわけである．

DevReplay の得意分野は GitHub プロジェクトの大半を占める小規模なソフトウェア開発である．小規模な故にあらかじめルールを網羅する手間はかけにくい．こういう場合に，自動的にどんどん修正ルールを蓄積していけるのはありがたい．これにより，ソフトウェア開発者は容易に巨人の肩の上に立てる．

現時点で DevReplay が扱えるのは，C, C++, Java, Dart, JavaScript, Python, Go, TypeScript, COBOL, Ruby, PHP, R の 12 言語である．今後も利用者の要望に応じて追加すると，上田君は意気込んでいる．

なお，DevReplay の応用として，上級者の書いたソースコードを正解集合として，開発者のソースコードを評価することで，教育に利用することができる．また，自分のソフトウェアと類似した機能を持つソフトウェアがセキュリティに関する対応を行った際に自らのソフトウェアのソースコードにも同様の対応をするように通知することができるようになる．

（竹迫良範 PM 担当）



■図-1　Visual Studio Code における利用例．左側は修正前のコード，右側は修正ルール

■大峠 和基（おおたお かずき），森 篤史（もり あつし）

SNS 向けテロップ自動生成[☆3]

私のようなオヂーサンとは縁遠い 10 代から 20 代の若者の間では，動画を投稿しあうコミュニケーションが盛んになっている．いまや，若者にとって動画を視聴する・投稿するということは身近なエンタテインメントとコミュニケーション手段の 1 つであり，若者には「格好いい・可愛い・面白い動画を作りたい」という大きな要求がある．そういえば，大峠君自身，ミスター筑波大学に選ばれ，森君はそのプロデューサだった．彼らは若い世代の感性を知り尽くしている．しかし，動画を作るには PC と動画編集ソフトの習得が必要である．

大峠君と森君の Telorain は，スマホだけで若者たちが撮った動画に，音声認識と構文解析をして気の効いた字幕（テロップ）をつけるサービスである．ポップな字幕は人気動画投稿者が動画の魅力を大幅に上げる常套手段だ．

Telorain の基本コンセプトは，(1) PC や動画編集ソフトに馴染みのない若者でも扱えること，(2) 若者に好まれる「SNS 映えする」デザインを簡単に作れること，(3) テロップ作成工程を自動化することにより，編集時間を減らすこと，である．

動画ではお見せできないが，図 -2 のような字幕が音声と同期して画面に出てくる．こだわったのは，字幕の文字のスタイルを SNS 映えするように多彩にしたこと（もちろん縦書きも可能），構文解析を行った上での改行位置の最適化である．

☆3　https://telorain.com/

もちろん雑音の多いところで動画を撮ることも多いので，誤認識の場合の編集機能もスマホ上でさくさくできるようになっている．競合の「撮るだけユーチューバー」では音声認識はするが誤認識だらけの字幕だし，字幕を手動で付ける InShot は編集機能は優れているものの手動の壁がある．実際，Telorain は字幕付き動画完成までの時間が半分以下ですむ．

Telorain の字幕自動生成の流れが分かるのが図 -3 である．簡単に書いてあるが，実はかなりいろいろなことを試した．単純なものでは笑い声検出による www www の挿入．複雑なのは，複数話者がいるときの話者分離と字幕配置の最適化である．これは技術的な関門が高く，まだ開発続行中とのこと．

Telorain は開発期間内に β 版を公開し，100 名を超えるユーザで評価を行った．その後，さらなる改良を加え，自動テロップ作成を 1 桁高速化し，脚註にあるページで AppStore での一般公開を行っている（iOS 版のみ）．このページを見ると，百聞は一見にしかずの，図 -2 の動画サンプルを見ることができる．そういえば，Telop＋Rain が名前の由来で，ロゴも雨傘をうまくあしらっているが，単に雨の日に名前を思いついたからとのこと．

大峠君はその後，相棒を代え，Telorain を未踏アドバンスト事業でさらに開発を進めることになった．いわく「ツール感をより減らして WOW 体験を増加させ，また世界各国でテロップ動画コミュニケーションを目指す」とのこと．まさに本気度の高い挑戦である．（五十嵐悠紀 PM 担当）

■大塚（おおつか） 馨（かおる）

組込みデバイス向けファザー☆4

未踏ではときどきすごい高校生を発掘することがあるが，大塚君もその 1 人（当時まだ 2 年生）である．飄々としているのに内にものすごい IT パワーを秘めている．

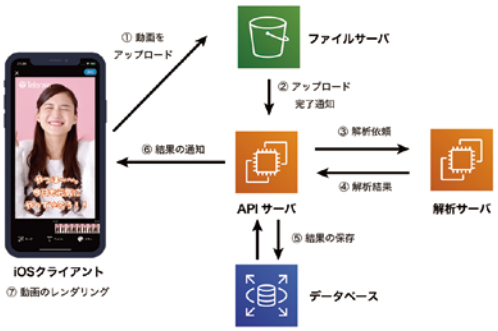
近年，脆弱性の発見の手法として，入力空間を網羅的に探索して脆弱性を発見するファザー（Fuzzer）が普及しつつある．もっとも対象となっているのはもっぱら Intel 製 CPU だったが，世の中にその 1,000 倍以上流通している ARM 製 CPU をベースとするシステムの脆弱性検査のニーズが高まってきた．実際，Apple や Google はバグ発見の報償金としてそれぞれ 100 万ドル，150 万ドルを支払うと公表している．

ファジングとは，対象としているシステムに，問題を起こしそうなデータ（fuzz）を入力し，システムの応答や挙動を監視する手法であり，応答に応じて自動的に怪しそうな fuzz をどんどん生成していく．こうしてプログラムのパスをなるべく多く通過するようにするのである．

大塚君はカーネルを対象としてファジングを行うために，最近発表された CoreSight というハードウェア支援トレースを使うことにした．これを使うとソースコードがなくてもブランチ命令を捕捉できる．ファジング対象となる仮想マシンの ARM をホストの ARM 上で動作させるために，改造 KVM（Kernel Virtual Machine）と改造 KVM lowvisor

☆4 https://github.com/roppinhoppin/kernel_crashes

■図 -2　作成されたテロップ付き動画の例



■図 -3　Telorain の字幕自動生成の流れ

（図 -4 の緑部分，ハイパーバイザの中の低レイヤモジュール）を開発した．改造 KVM lowvisor は CPU のハードウェア支援機能を使い，仮想マシンの実行を高速化する．これに CoreSight を組み合わせることで高速にゲストマシン内のカーネルやアプリケーションのファジングができる ZeroSight と名付けたシステムが機能するはずであった．

ところが 7 種類の ARM 搭載 CPU ボードを試したが，どれも CoreSight の実装が不完全なことが判明した．そこで，プロセッサエミュレータ QEMU を改造してソフトウェアトレースによるファジング機能も並行して開発し（図 -5），使用可能な CoreSight が入手できれば差し替えるという戦略をとった．

代替 ZeroSight を ARM 版 Linux カーネルに適用して，9 日間でなんと 326,881 個のバグを検出した．この中には深刻な 10,781 個のクラッシュバグが含まれる．これにより 69 個の異なる基本ブロックに含まれるバグが検出できたことになる．

試用した CPU ボードを焦がしたりしながらの悪戦苦闘の結果，ここまでのことを成し遂げたのは素晴らしい胆力だ．本人も人間的に成長できたと自覚したようだ．ZeroSight は 2016 年度未踏スーパークリエータ木村廉君が起業したリチェルカセキュリティの脆弱性検査ツール「Wyvern Pro」に採用される予定である．（首藤一幸 PM 担当）

■岸田 聖生（きしだ しょうき）

電気を肌感覚で理解できるツール☆5

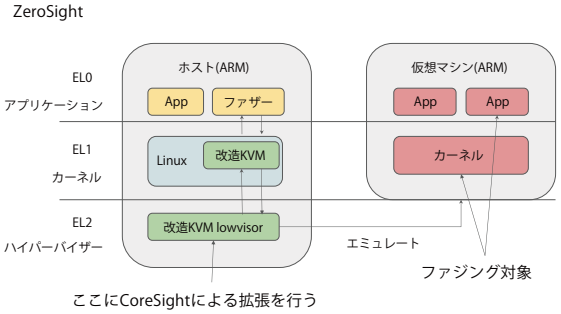
我々にとって電気は日常生活に欠かせないものだが，電気にかかわる法則を肌感覚で理解している人は少ない．感電はまさに肌感覚だが「法則の理解」にはつながらない．

岸田君は，家庭や学校に置くにはちょっと大仰だが，肌感覚で電気の法則を理解できる装置を作り上げた（図 -6）．装置の名前は Ambre（Augmented Metaphor Based Representation system for Electricty），まさに電気の語源となった琥珀（ギリシャ語）である．実は未踏の前に外見が似た装置を作っていたが，ほぼ完全に作り直した．

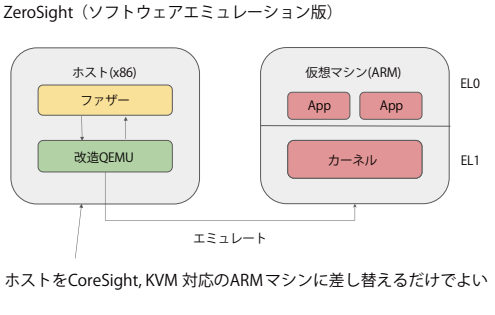
Ambre の上面はいわば巨大なブレッドボードで，その上に 2 段重ねにできる電池，結線を行う配線デバイス，電圧に応じて光る LED，モータなどを置く．配線デバイスの線は弾力性のあるシリコンで構成されており，握ると断面積が小さくなり，まるで電気の流路が本当に狭まるように配線自体の抵抗値が上がる．また，電流が流れるとその量に応じてブルブル震えたり，試験的には内部の LED リボンが流れるように光る機能も開発した．

図 -6 の両側にある上下に動くアクチュエータは横 4 列ある接点の電圧を高さで表現している．開発中，電気的に浮いている接点に対応するアクチュエータが上下に不安定に動くのはなんともリアルだった．

☆5 https://ambreio.net/



■図-4　当初予定していたZeroSight



■図-5　代替のソフトウェアエミュレーションZeroSight

図-7 はこのアクチュエータの部分をケースを外して撮ったものである．

表には出ていないが，彼の成果報告書を読んで驚くのは，ものづくりに関する町工場の職人さんのような徹底的なこだわりである．Ambre は全部試行錯誤を経た手作りなのだ．デリケートな Ambre の持ち運びケースも自分で作った．このほかにも，コンデンサ，交流の周波数表現にも取り組んだが，岸田君のこだわり感覚ではまだ表に出すレベルではないようだ．

この装置をただ作るだけではなく，何回もワークショップを行い，教育効果の確認，フィードバックを得た改良に取り組んだ．中でも，ユニットを置いてから実際に電気回路としての反応が現れるまでの時間が 1.6 秒だったのを 0.2 秒にした改良は大きい(反応が遅いと，子供はすぐユニットを動かしてしまう)．これには開発環境の丁寧な見直しが必要だった．ワークショップに参加した 30 歳男性の「最初からこうやって習いたかった」というコメントは実感がこもっている．また，子供たちが直列・並列の差を実感として自ら発見したのには感動したとのこと．

岸田君は Ambre の改良（特に交流対応）をさらに続け，各地の科学館での展示に活用することを考えている．たしかにそれに相応しい完成度にすでに達しつつある．

私はこれを見て，あの落合陽一君の 2009 年の未踏プロジェクト「電気が見えるデバイスとソフトウェア」を思い出した．そういえば，岸田君にも落合君に似た「考える人」の風格がある．

（田中邦裕 PM 担当）

■久野 文菜（くの あやな）

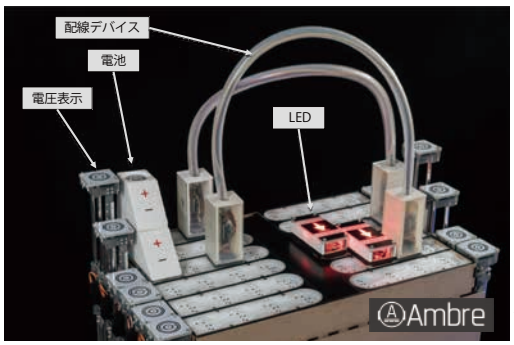
多重奏からの単一楽器聴音支援☆6

久野さんは大学のビッグバンドサークルの中でジャズピアノを弾くアマチュア演奏家である．そんな彼女の最大の悩みは，有名な曲でもジャズの楽譜にはコード進行以外は何も書かれていないこと．だから，良い演奏の CD などを聴いて，どう弾いているかを調べなければならない．これを耳コピ（採譜）という．演奏をたしなむ者はすべからく耳コピの能力が必要なのだが，単一楽器の演奏ならともかく，ビッグバンドのような大編成の演奏から自分が耳コピしたい楽器を聞き分けること（音源分離）がそもそも難しい．

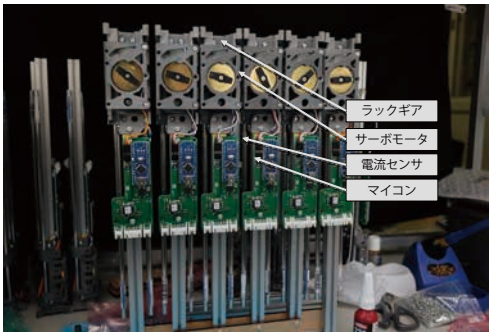
演奏仲間からもエールを受けて挑戦したが，彼女のオリジナルの手法は音のスペクトログラムを画像として学習して合成音から単独楽器音を分離することだった．しかし，スペクトログラムに乗った画像ノイズに悩まされ，そこからなかなか進まない．

そうこうしているうちに，2019 年 9 月，音の時系列分析に強い Bidirectional LSTM（深層学習の 1 手法）を使った Open-Unmix というソフトがオープンソースで公開されてしまった．ここで PM と相談の上，目的を達成することを優先したのが功を奏した．Open-Unmix のコントリビュータになるくらいに調

☆6 https://www.musep.net/



■図-6 Ambre の上面



■図-7 電圧表示アクチュエータの並び

べあげてそれに乗ったのである．

丸１カ月かけて，音源分離に必要な学習データ（ドラム，ウッドベース，ボーカル，ピアノ）をKRONOSという電子楽器を使って各300時間分（！）用意した．こうした苦労の上作り上げたのがiOSとサーバの組合せで動くMuSep（ミューゼップ，Music Separator）である．耳コピをするためのスマホの画面を図-8に示した．４種類の楽器のうちの１つを選択して，聞きたいところを速度調整をしながら，かつ全体の音も適当にミックスしながら聞くことができる．さすがに採譜まで自動化しているわけではない．人間の能力に頼ることも必要だ．

ここまでできると，逆に自分が演奏したい楽器の音だけを消して，有名ビッグバンドのピアニストになった気分での即興演奏をしたくなる．MuSepはそんな自動伴奏（？）音源を作ることも可能である（図-9）．

久野さんは前向きの心と，素晴らしい気迫を持って，音源分離をかなりの完成度まで仕上げた．競合アプリに十分勝てる．PMや同期のクリエータたちの助言も効果的だった．MuSepは県のサポートを受け，テキサス州での大イベントSXSWに行けることになっていたが，新型コロナのために来年に持ち越すことになった．また，JBMC（ビジネスコンテスト）の日本大会でも優勝した．音源分離はやさしい問題ではないが，この勢いで実用化に突進してほしい．

ところで図-8のあちこちに可愛い犬のイラストがあるが，これはすべて久野さんのオリジナルである．

（首藤一幸PM担当）

■櫻井 碧（さくらい あお）

生命情報解析向け秘密計算クラウド☆7

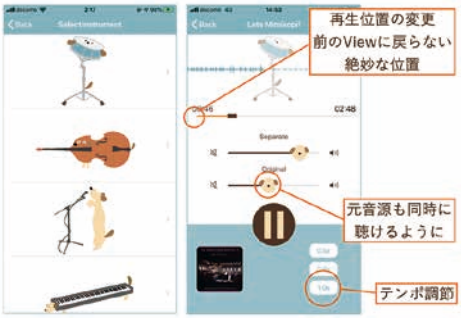
近年，生命情報解析の分野におけるクラウド利用が活発である．しかし，生命情報のような機密性の高い情報を用いてクラウド上で計算を行う際には，漏洩に対して安全であることが客観的に保証できる手段，すなわち「秘密計算」を使って処理することが必要である．

準同型暗号に並んで秘密計算としていま注目されているのが，信頼できる実行環境（TEE, Trusted Execution Environment）の利用である．TEEの実例としてIntel社のSGX（Software Guard Extension）がある．これはIntelのCPUの拡張機能で，RAM上に暗号的に厳重に保護された小区画（Enclave）を生成し，その中で計算を行う方法である．

ここまではいいのだが，Intel SGXの最大の欠点は，その超弩級の使いにくさである．ちょっとしたコードを書くのに四苦八苦せざるを得ない仕様なのだ．これは生命情報の研究者には敷居が高すぎる．

櫻井君はこのSGX伏魔殿を研究者が簡単に使えるようにすることに挑戦した．未踏関係者がみんな心配するほど憔悴しながらの開発となったが，なんと，たとえば遺伝子解析の定番であるGWAS（ゲノムワイド関連解析）のSGXの（遠隔認証などの処理を含めた）４万行のコードをたった４行で書けるようにしてしまったのである．さながら伏魔殿に果敢に飛び込んだ救世主だ．

☆7 https://bi-sgx.net/



■図-8　MuSepで音源分離をして耳コピをする画面

■図-9　ビル・エバンスに代わって即興の練習

話が前後したが，櫻井君が開発した BI-SGX（BI は Bioinfomatic からきている）の概要を図 -10 に示す．秘匿したい生命情報の所有者がデータを送信すると所有者の Enclave で暗号化（シーリング）して保持する，研究者が計算を行う Enclave には復号してそのデータを渡す．研究者が送ったコードは Enclave 内で実行され，その結果が返却される．研究者が受け取るのは統計的な結果なので，個別の元データの詳細を知ることはできない．

研究者が 1 個のデータの平均を計算すれば個別の元データを知ることができるが，櫻井君の「インタプリタ Qliphoth」がそういう計算ができないガードをする．つまり，櫻井君は有用な生命情報計算を行うインタプリタ言語を作ったのである．行数が短く書ける代わりに，データの詳細にかかわる計算ができないようにしたのだ．このインタプリタ手法によりサイドチャネル攻撃にも耐性ができた．先行研究の PRINCESS，Graphene-SGX，SGXElide，そして SGX-BigMatrix のいずれよりも性能的・機能的に優れたものになった．

また，計算分野ごとにインタプリタを作る手法自体が汎用的なので，この手法が他分野にも普及することを期待したい．なお, BI-SGX は一部のパブリッククラウド上でも駆動できる（表 -1）．

未来指向の研究として盛んなのは準同型暗号だが，現時点では SGX のような TEE を使うほうがデータ量や計算時間で数桁以上のアドバンテージがある．

（藤井彰人 PM 担当）

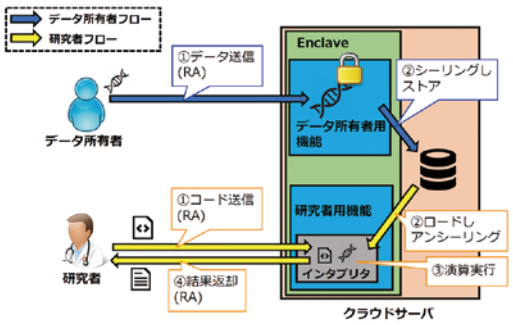
■ 田脇 裕太（たわき ゆうた）

介護予防のためのリハビリ支援

平均寿命とは別に健康寿命という概念がある．つまり，なんらかの介護が必要になる手前が健康寿命で，世界的に見てその差は 10 年という（日本人は男女とも 70 歳代半ばより前が健康寿命）．だから，自立して生活できる健康寿命の延伸が高齢化社会の重要な課題となる．介護施設では，主に要支援・要介護者を対象としてリハビリ（機能訓練）を実施しているが，多くの介護施設では，設備もノウハウも十分に整っておらず，リハビリの最適化はおろか，利用者のデータ収集さえも十分にできていないのが現状である．

田脇君は実際に介護施設の現場で研究しながら，この問題について造詣を深め，解決法を探ってきた．そこで得た知見から未踏の明確な挑戦課題を定めた．提案書に面白いことが書いてあった．「医工連繋」はよく言われるが「介工連繋」はまだない．これを実現するのだと．

当初，靴に仕込んだ加速度センサなどで歩行の様態を IoT 的に分析していたが，これでは介護全体に大きなインパクトをもたらせないことが判明した．そこでもっと広く考え，業務効率化とパーソナライズされたリハビリを両立させるシステムと，収集したデータに基づいて健康状態の遷移を可視化するリハビリマップを開発することになった．それが功を奏した．



■図 -10　BI-SGX を用いた秘密計算

■表 -1　BI-SGX が使えるクラウド

クラウドサービス名	SGX対応のマシンがあるか？	SGXが有効化されているか？	BI-SGXが利用可能か？
Microsoft Azure	○	○	○
IBM Cloud	○	○	○
Alibaba Cloud	○	○	○
AWS	○	×	×
Google	○	×	×

実際に実現したことは，(1) 患者の健康状態から適切なリハビリメニューを提案して顧客満足度を上げることと（図 -11），(2) 提案内容のエビデンスとして，患者の状態遷移を可視化するリハビリマップを設計したことである（図 -12）.

図 -11 を見て分かるように当初の靴装着型センサのほかにスマホを用いて各種のデータを収集し，それをサーバで分析し，患者にパーソナライズしたレコメンドを提携リハビリ施設の理学療法士に伝える．たとえば，リハビリモデルに基づき，筋力が実年齢よりも衰えているので膝筋力に関係するリハビリを提案したり，類似度に基づいて「A さんと同じような状態の人は，XX というリハビリをしていますよ」といった提案を行う.

図 -12 のリハビリマップはプロジェクト期間中に，なんとバージョンを 5 つも重ねた．最終的に富士山登山をメタファーにしたとても分かりやすい 3 次元的なマップになった.

田脇プロジェクトの強みは実際に介護施設と連繋しながら開発を続けられたことである．だから絵空事ではない着実なシステムとして仕上がった．さすが富山湾 47km 横断遠泳を成し遂げたガッツだ．この成果をベースに世界で年間 1,300 万人以上が発症するという脳梗塞患者のリハビリに向けた，国際的に通用するシステムを未踏アドバンストで開発することになった. 健闘を祈る．　（稲見昌彦 PM 担当）

■中野 萌士（なかの きざし），堀田 大地（ほりた だいち）

VR 空間における食体験☆8

VR（実質的現実）元年が何年も続いているが，HMD（ヘッドマウントディスプレイ）を被ることに違和感がなくなってきたことは事実だ．VR 空間では現実ではできないことが可能になるが，難しいこともある．その代表例が食事だ.

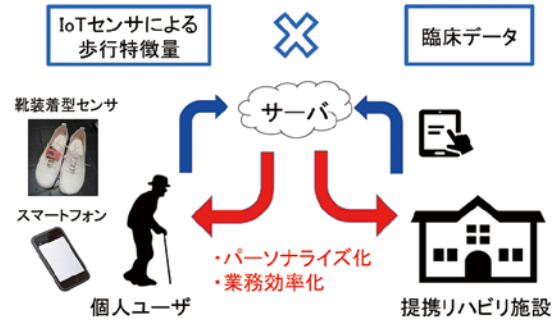
何かを食べることは身体性にかかわることなので，HMD ではそう簡単にはごまかせない．しかし，中野君と堀田君は果敢にこの問題に挑戦し，Ukemochi（ウケモチ）というシステムを作った.

VR における食体験の操作は，(1) いま食べているものを空想の世界での食事に錯覚させることと，(2) いま食べている食事を異なる食事に錯覚させることに大別されよう.

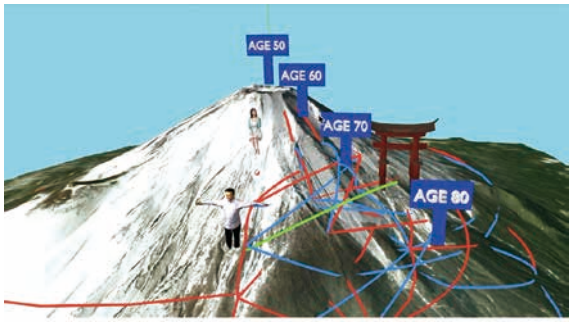
たとえば，図 -13 に示したのは桜の名所に行った気分で，自分の部屋でいま現に食べているカツ重を味わっている場面である（本来は動画）．図 -14 はバーチャルな「彼女」と一緒に食事しているところである．右上は相手から見た（なぜかボーイッシュな女性の）自分である．自分で買ったチョコを VR の中で彼女にもらったと想像しながら食べることによって味わい深いものにすることも可能だ．もちろん，自分が食べているものは HMD のカメラで撮っている.

上記の（2）に関しては，いま食べている安い豚肉を，形状を保ったまま上級の霜降り肉に置き換え

☆8 https://signs0302.github.io/ukemochi/



■図-11　システム構成図



■図-12　リハビリマップ

るとか，現実には存在しないドラゴンの肉をゲームで倒したあと焼いて食べるといった未体験の食事を提供することも可能になる．いま食べているスルメをドラゴンの味に錯覚させるのだ．

しかし，これを実現するには難しい映像処理が必要になる．自分が食べているものを（食器も含めて）映像から安定的に切り出し，それを HMD 内の映像の正しい位置に重畳し続けないといけない．そのため，彼らは物体検出と物体追従の技術を組み合わせた．

物体，つまり食べているものの検出には YOLOv3 という検出方式を使った．これは後述の物体追従器が追従すべき物体を較正するために必要な初期値座標を推論する．YOLOv3 は解像度 320 × 320 の入力に対して，22ms（45FPS）で推論を行うことができる．食べているものの追従には SiamMask というソフトを用いた．これは違和感なく VR アプリケーションへ重畳するための食事領域とその座標を出力する．これにより，食べているものの 3D モデルを重畳することが可能になる．

実際は，YOLOv3 の解像度から見ても正確な重畳が行えているわけではない．デモ映像の不安定さは隠しようがないが，なにしろ VR 空間における食体験という発想自体が未踏的であり，彼らの素晴らしい業績である．実際，Ukemochi を紹介した動画は 2 週間で約 6 万回再生された．

なお，彼らは同じ指向であることをリモートで発見し，手を組み，そのままリモートで開発を続けた．まさに新コロナ時代のリモートワークの先駆的な共同開発だった．彼らは性格的に相補的でバランスが取れていたようだ．（稲見昌彦 PM 担当）

■松浦（まつうら） 知也（ともや）

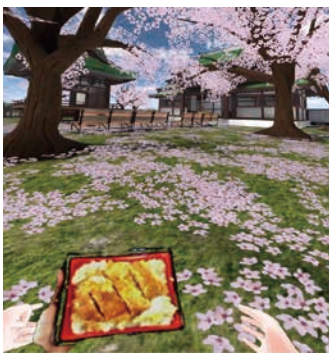
プログラマブルな音楽制作ソフトウェア☆9

音楽に特化したプログラミング言語は 1950 年代から始まり，派生やクライアントを含めるとこれまで 30 種類も開発されてきた．最近は音楽信号処理のような低レイヤ処理から，楽譜のような高レイヤ処理までを見据えた言語の開発が増えてきている．

松浦君はこれらの処理を美しく設計された 1 つの言語体系の中で完結でき，かつ実時間処理に耐える高速な言語 mimium（MInimal Musical medIUM）を開発した．しかし，波形レベルから楽譜レベルまでを統一的に記述することは容易ではない．松浦君いわく，mimium は「音楽は音楽家が，それ以下の信号処理のような低いレイヤの処理はエンジニアがといった，異なる興味の対象を持つ人同士をつなぎ合わせる役割を持っているインフラストラクチャ」である．

mimium は過去のいつくかの有名言語 Max, Faust, SuperCollider, Extempre などを参考に現代的な設計を行った．言語仕様の詳細をここで紹介することはできないが，信号処理レベルでは音の振幅を出力する関数が直前に返した値を self という特殊な変数で参照できるようにしているのが面白い．関数が直前の履歴を持っているのだ．また，c(b(a(x))) とい

☆9 https://github.com/mimium-org/mimium



■図-13　バーチャルな桜の名所に出かけて，カツ重で花見

■図-14　憧れの彼女とディナー

う関数呼出しの入れ子は x |> a |> b |> c のようにパイプライン演算子 |> を使って書くことができる．Max に慣れたユーザだったら嬉しい記法だ．

楽譜レベルの遅い処理では関数呼出しのタイミングを @ 記号で示すことができる．たとえば，

Nloop(period)@(now＋nextperiod)

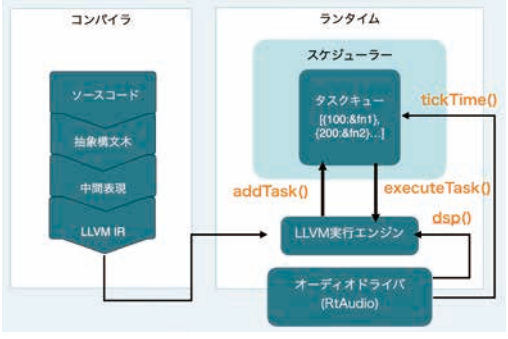
と書くと，特殊変数 now が持っている現在時刻に nextperiod を足した時刻に Nloop(period) という（音を鳴らす）関数を呼ぶというわけである．従来の言語では callback などを使って書いたがこちらのほうがずっと見やすい．ただし，関数の返り値をどうするかは非同期処理の面倒な問題で，これは変数の破壊的代入などの副作用を伴う手続き型の処理で逃げている．図 -15 は mimium で書いた波形処理レベルのプログラム例である．dsp は毎秒 48,000 回実行される特殊な関数なので，これは 440Hz のこぎり波を鳴らす．右にあるのは横断歩道をイメージした mimium のロゴである．

mimium をライブコーディングのような実時間演奏に使うためには処理の高速化が必須である．そのため LLVM を利用した（図 -16）．コンパイラとランタイムは主に C＋＋ を用いて実装した．これは今後の拡張を見据えたアーキテクチャである．今後はデータ構造の増強，マルチ OS への対応を行う．普及のための統合プログラミング環境の整備も予定している．

成果報告会ではちょっとシュールな電子音楽を聞かせてくれた．（竹迫良範 PM 担当）

```
fn phasor(freq:float){
return (self+freq/48000)%1
}
fn dsp(time:float)->float{
return phasor(440)
}
```

■図-15　miminumのプログラムとロゴ



■図-16
mimium の
アークテチャ

■松岡 航太郎（まつおか こうたろう），伴野 良太郎（ばんの りょうたろう），松本 直樹（まつもと なおき）

準同型暗号によるバーチャルセキュアプラットフォーム☆10

秘匿しないといけないデータに秘匿しないといけないプログラムを適用し，第 3 者がその途中を見ても有意な情報が得られない計算を「秘密計算」という（前出の櫻井君の項を参照）．秘密計算のうち，数学的に最も安全性の高い方式が準同型暗号である．これは，データもプログラムも暗号化した状態で計算を行った結果を復号したものが，暗号化していない平文データと平文プログラムでの実行結果と一致するという性質を持つ計算手法である．図 -17 は松岡君たちが開発したバーチャルセキュアプラットフォーム（以下 VSP）の概要図だが，これ自体が準同型暗号の仕組みを表している．櫻井君の BI-SGX は，万が一 CPU の製造元が Enclave に何かを仕掛けたら秘密が洩れる．準同型暗号はそういった心配がない（もちろん暗号強度が信頼できるかぎり）．

準同型暗号には演算種類が限定されるものが多いが，任意の演算が可能なもの，つまり任意の論理回路を暗号化したまま計算できるものを完全準同型暗号という．松岡君たちが扱うのはもちろん完全準同型暗号で，TFHE と呼ばれるものだ．

これは素晴らしいのだが，安全性の高さの代償が速度とデータ量である．いまのところこれを高速に実行する手段はない．しかし，速度やデータ量はと

☆10 https://github.com/virtualsecureplatform/kvsp

もかく，まずちゃんと動くものを作ろうというのが松岡君たちの VSP の挑戦だった．目標は C のプログラムを完全準同型暗号で実行できることである．

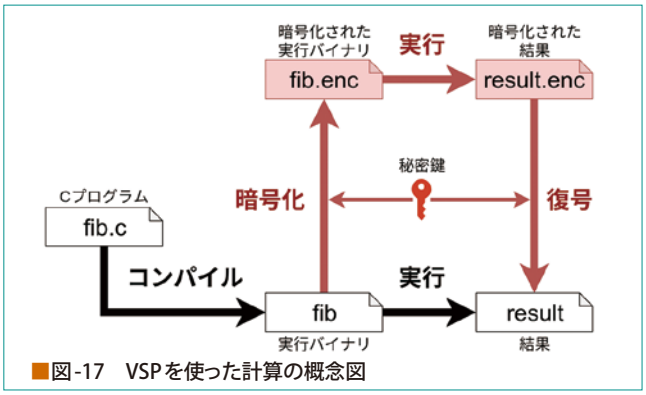
松岡君が準同型暗号の勉強から始め，VSP のエミュレーションすることになる CPU の設計と実装（担当松本君），そこへ C をコンパイルするコンパイラ（担当伴野君），CPU での計算を回路の論理素子のレベルまで暗号化したまま実行するエミュレータ（全員）を驚くほど何回も作り直しながら仕上げていった．たとえば CPU の命令セットは 32 ビット，16 ビット，16/24 ビット混合と変遷，それに応じてコンパイラも，エミュレータも作り直し……．作り直しの主な動機は性能向上である．ゲートの動的並列実行にも注力した．ともかく性能向上イノチの開発を続けたのである．しかも，各種の検証システムを利用し，システムとしてバグながないことに常に意識を払った．これらの努力は限られたスペースではとても紹介しきれない．

結果としてできあがったのが KVSP（Kyoto VSP）である．ソフトウェアなので，図 -17 以上に分かりやすいコンパクトな図を追加できないが，KVSP の特徴をちゃんと紹介すると，標準ライブラリを必要としない範囲の C プログラムを，プログラムを秘匿したまま実行することができる，ソフトウェアのみで実現可能な実行基盤の PoC（Proof of Concept）を与えた世界初のシステムである．

現時点の速度性能は，CPU エミュレータは，NVIDIA Tesla V100 を 1 基搭載したマシンで 1 クロックあたり 1.7 秒，8 基では 0.8 秒である．また AWS の c5.metal インスタンスでは 1 クロックあたり約 2.3 秒で動作する．メモリは 512 バイト．黎明期コンピュータ並みの信じられない遅さだが，彼らは将来これが「プログラムの秘匿」が無二の価値であって人々を惹きつける限り，VSP の性能向上の試みは続くと考えている．だから，VSP は数十年後に普及し得ると……．

3 人とも学部生，ともかく元気で好奇心旺盛，未踏でこんな遠い先を見据えたプロジェクトは初めてだと思う．ともかくこの分野でしっかりエンジニアリングした世界初の成果であることは間違いない．

（首藤一幸 PM 担当）



■図 -17　VSP を使った計算の概念図

■森 瑞穂（もり みずほ）

高速な VMI 機構を実装したバイナリ解析基盤☆11

コンピュータシステムに悪さをするマルウェアが後を絶たない．マルウェアは年々巧妙化しており，検出が難しくなってきている．通常，マルウェアを動かしながら解析するときは，たとえばハイパーバイザの上のゲストマシン上で走らせ，それを監視しながらログを取る．しかし，ハイパーバイザとゲストマシンの間のコンテクストスィッチ（CS）が多く発生し，どうしても時間がかかってしまう．高速な解析のためには CS を減らす必要がある．森君が開発した FastVMIX（Virtual Machine Introspection on Intel VT-X）のアイディアは，解析時にゲストからハイパーバイザに制御が移る CS をなくし，解析自体をゲスト側に移してしまおうというものだ．

一見，それではマルウェアと解析器が仮想マシン上で同居することになってしまい，逆に危なそうだが，以下に述べる巧妙な手法でそれを交わす．図 -18 に全体図を示す．

図 -18 の左，Ring 0 で動作する PIC（Position Independent Code）バイナリがゲストのセンシティブな動作をトラップし，ログを取る．ゲスト

☆11 https://github.com/morimolymoly/fastvmix

内部（Ring0 または 3）に侵入したマルウェアはシステムを攻撃することが可能になるが，BitVisor ベースのハイパーバイザがページテーブル保護とメモリマップの動的変更を行うことによりシステムを保護する．

まず，マルウェアが PIC バイナリやログをアンマップしたりすることを監視する．そのオーバヘッドを大幅に削減するのが，1GB Huge Page の採用である．つまり，どでかいページにすることにより，ページテーブルエントリのチェックをわずか 2 レベルで済ませることができた．

次に，図 -19 に示すように，通常のメモリマップと解析用のメモリマップの 2 種類を用意し，解析時のみ解析用のメモリマップ（PIC バイナリがマップされ，ログが読み書き可能，カーネルのコード領域は読み込みのみ）を使用するのである．メモリマップにはゲストの物理メモリアドレスとホストの物理メモリアドレスの変換テーブルである EPT（Extended Page Table）を用いる．メモリマップの動的変更には EPTP スイッチングを用いて，コンテキストスイッチなしでメモリマップの動的変更が行えるようにしたところがミソである．

また，最近の小賢しいマルウェアは監視されていることを，実行時間の差を見ながら検知する機能を備えている．これにはマルウェアの CPU 時間を偽装することで応じる．タイムカウンタを取得する RDTSC 命令をトラップし，解析にかかった時間（実はもう少し安直な固定値）を引いて返す仕組みをハイパーバイザに実装した．ベンチマークプログラムではこれが有効に働いていることが確認できた．

こうした工夫の結果，先行システムである DRAKVUF よりも 306 倍（！）高速化することができ，ネイティブコードより 5.3 倍遅い程度になった．これにより，インシデントの発生時に速やかに対応が行えるようになるだろう．

FastVMIX はオープンソースとして公開されるので，さらなる改良の動きが広がることを期待したい．

（田中邦裕 PM 担当）

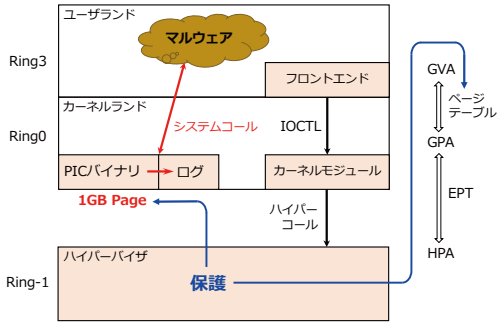
■森山 多覇（もりやま たは）

指先の触覚を背中に転移させるデバイス

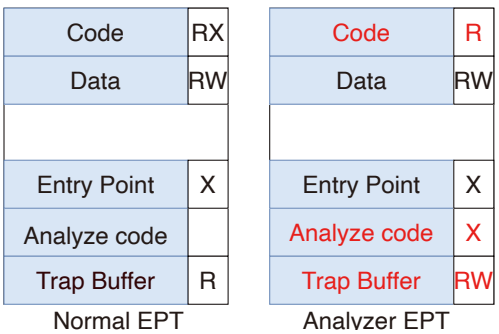
VR 環境の中で指に触覚情報を提示する研究は盛んに行われているが，指につけるには大きすぎる装置になってしまう．最も繊細な動きができる指の運動に干渉してしまうのだ．

ならば，指が感じる触覚情報を体の他の部位に転移，つまり「触覚転移」させたらどうか．これが森山君の基本アイディアである．以前から森山君は触覚転移の研究をしていたが，未踏では体の中で最も面積が広い背中を利用することにした．背中の 2 点弁別閾は 50mm，とても粗いのだが，面積が広いので，144 個の振動子をベスト（Vest）の背中部分に規則正しく並べれば，指先の触覚と同等の解像度を得ることができる．しかも，背中はどんなときでも空いている．

開発は 2 つの異なるデバイスに結実した．1 つは



■図-18　FastVMIX の構造



■図-19　メモリマップの動的変更

当初から計画していた，VR 空間での指の触覚を背中に転移する Simple is Vest である．これは Leap Motion を用いて指先をトラッキングし，Unity を用いて VR 空間を構築する．Unity 上に表示されている指腹に，等間隔に 144 点の接触点を設置し，物体に触れた際に，Unity で接触判定を行い，空間的に対応する背中の振動子を駆動する．個々の振動子の振動強度が一定で，振動時間しか制御しないので，Simple is Vest は安価だが，単調な触覚情報しか得られない．VR 空間ではそれでも十分である．

当初の計画はここまでだったが，秋ごろにできてしまったので，森山君はそこからさらなる挑戦をした．その成果が，現実空間での指の触覚を背中に転移する HARVEST である（図 -20）．まず振動子を応答性の高いフォースリアクタに変更した．これで個々の振動子を異なる振動数・振幅で制御できるようになった．またグローブ型センシングデバイスを用いて指の接触点をセンシングする．これは親指と人差指の先端にそれぞれ 2mm × 2mm 解像度の圧力センサを配置したグローブである．144 個の振動子には少し工夫をしたマッピングを行っている．

実は 144 個の振動子を制御するのは容易ではないが，マイクロコントローラ 1 台とシフトレジスタだけで済ませているので，デバイスとはとてもコンパクトになる．実際，HARVEST を着てみて，服としての違和感はまったくない．ただし，背中に密着するようにピッタリと着用する必要がある．グローブをはめて指先でいろんなものを触ってみると，背中になんともゾワゾワした感触が走る（ムシズが走る？）．しかし，これは慣れると本当に指の繊細な感触が表現されていることが分かるようになる．

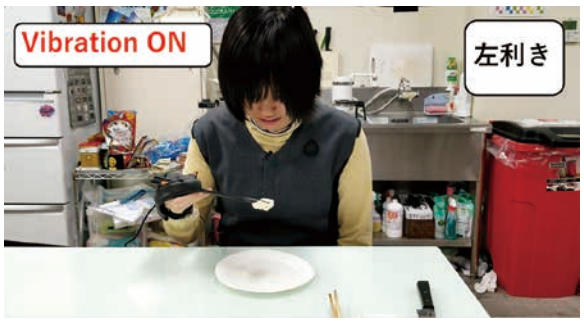
ここからがこのプロジェクトの真骨頂だと思うのだが，森山君は HARVEST を使って面白い実験を行った．最初の実験は，被験者 4 人に「目を閉じて，利き腕で箸を使用し，絹ごし豆腐をつまみ，持ち上げてもらう」ことである．振動を与えないと 4 人全員が柔らかい豆腐を持ち上げることができなかったが，背中に触覚転移の振動を与えると 4 人中 3 人が持ち上げられるようになった．

さらに面白いのが「目を閉じて，利き腕とは逆の手で箸を使用し，絹ごし豆腐をつまみ，持ち上げてもらう」実験である．左利きの人に右手で豆腐を持ち上げてもらったところ，HARVEST を使ったら持ち上げることができた（図 -21）．これは指先から得ることができる微細な振動刺激を，背中などの別部位に転移することで，指先の触知覚能力を強力に支援できるということを意味している．

在外生活の長かった森山君は実に多彩な能力を持った人で，プロサッカー選手になるべく 4 年間ドイツのサッカークラブに所属，エジプトでの国際バイオリンコンクールで 2 年連続 2 位などなど枚挙にいとまない．すごい人だ．（稲見昌彦 PM 担当）

竹内郁雄（正会員）■ nue@nue.org
1971 年東京大学大学院修了．以降，NTT 研究所，電気通信大学，東京大学，早稲田大学を経て現在，IPA 未踏 IT 人材発掘・育成事業統括プロジェクトマネージャ，一般社団法人未踏代表理事．（株）Givery 技術顧問，東京大学名誉教授．本会フェロー．

■ 図 -20　HARVEST，一番右側が布をめくったら見える振動子の配列



■ 図 -21　目を閉じて利き腕と逆の手で絹ごし豆腐を持ち上げる