Original Paper

Quantification of Digital Body Maps for Pain: Development and Application of an Algorithm for Generating Pain Frequency Maps

Abhishek Dixit, BSc (Hons), MRes; Michael Lee, MBBS, FRCA, PhD, FFPMRCA

Division of Anaesthesia, Department of Medicine, University of Cambridge, Cambridge, United Kingdom

Corresponding Author:
Abhishek Dixit, BSc (Hons), MRes
Division of Anaesthesia
Department of Medicine
University of Cambridge
Box 93, Addenbrooke's Hospital, Hills Road
Cambridge, CB2 0QQ
United Kingdom
Phone: 44 1223217 ext 889
Email: ad825@cam.ac.uk

Abstract

Background: Pain is an unpleasant sensation that signals potential or actual bodily injury. The locations of bodily pain can be communicated and recorded by freehand drawing on 2D or 3D (manikin) surface maps. Freehand pain drawings are often part of validated pain questionnaires (eg, the Brief Pain Inventory) and use 2D templates with undemarcated body outlines. The simultaneous analysis of drawings allows the generation of pain frequency maps that are clinically useful for identifying areas of common pain in a disease. The grid-based approach (dividing a template into cells) allows easy generation of pain frequency maps, but the grid's granularity influences data capture accuracy and end-user usability. The grid-free templates circumvent the problem related to grid creation and selection and provide an unbiased basis for drawings that most resemble paper drawings. However, the precise capture of drawn areas poses considerable challenges in producing pain frequency maps. While web-based applications and mobile-based apps for freehand digital drawings are widely available, tools for generating pain frequency maps from grid-free drawings are lacking.

Objective: We sought to provide an algorithm that can process any number of freehand drawings on any grid-free 2D body template to generate a pain frequency map. We envisage the use of the algorithm in clinical or research settings to facilitate fine-grain comparisons of human pain anatomy between disease diagnosis or disorders or as an outcome metric to guide monitoring or discovery of treatments.

Methods: We designed a web-based tool to capture freehand pain drawings using a grid-free 2D body template. Each drawing consisted of overlapping rectangles (Scalable Vector Graphics <*rect*> elements) created by scribbling in the same area of the body template. An algorithm was developed and implemented in Python to compute the overlap of rectangles and generate a pain frequency map. The utility of the algorithm was demonstrated on drawings obtained from 2 clinical data sets, one of which was a clinical drug trial (ISRCTN68734605). We also used simulated data sets of overlapping rectangles to evaluate the performance of the algorithm.

Results: The algorithm produced nonoverlapping rectangles representing unique locations on the body template. Each rectangle carries an overlap frequency that denotes the number of participants with pain at that location. When transformed into an HTML file, the output is feasibly rendered as a pain frequency map on web browsers. The layout (vertical-horizontal) of the output rectangles can be specified based on the dimensions of the body regions. The output can also be exported to a CSV file for further analysis.

Conclusions: Although further validation in much larger clinical data sets is required, the algorithm in its current form allows for the generation of pain frequency maps from any number of freehand drawings on any 2D body template.

(JMIR Form Res 2022;6(6):e36687) doi: 10.2196/36687

KEYWORDS

Scalable Vector Graphics; SVG; pain drawing; pain location; Body Pain Map; overlap computation; heat map; pain frequency map; algorithm

XSL•FO
RenderX

Introduction

Background

Pain is an unpleasant sensation signaling potential or actual injury to the body site [1]. The location of pain can be communicated or recorded by drawing onto a body template (eg, Brief Pain Inventory [2]). Pain drawings have been used clinically and in research for decades [3]. Jang et al [4] reported that patients were more confident about communicating the locations of pain to the clinician in the form of a drawing as opposed to a written description, and the clinicians also favored drawings over written descriptions. Digital technology is now commonplace and circumvents problems associated with the processing and storage of pain drawings; hence, pain drawings are now widely acquired as digital images when possible [5]. Numerous web-based applications and mobile apps offer digital pain manikins, both commercial and academic [6]. Most digital pain manikins use 2D templates, which are whole-body coronal or sagittal views of the human body. There is often a choice of gender and body type for 2D or 3D surface templates [7-9]. A typical digital pain manikin consists of a body template and self-explanatory instructions for the patient regarding how to indicate the locations where they experience the most discomfort.

The pain frequency map is generated by the simultaneous analysis of all digital pain drawings to compute the locations of pain that participants have in common. To aid visualization, the maps also use color codes to highlight locations on the body in accordance with their frequency of occurrence in the sample being studied. Such maps are clinically useful in identifying where sensations commonly relate to disease anatomy [10,11] and the factors that influence the subjective localization of pathology. These maps are perhaps most important in chronic primary pain syndromes [12], which are defined by the locations of pain in the absence of disease (eg, chronic back pain).

The ease with which pain frequency maps are generated critically depends on the nature of the body template used. For body templates where the anatomical locations are already predefined and demarcated [7,13], the pain drawings are swiftly completed, and the data captured are binary, where *1* indicates a selection, and *0* indicates no selection. As a result, the generation of pain frequency maps is relatively straightforward, because only the number of participants who selected each location needs to be deduced [14]. However, such frequency maps may not entirely capture the underlined spatial patterns of pain because of loss of spatial resolution caused due to the pixel (ie, x-y coordinate, representing the smallest possible division for the body template) being preassigned to larger (hence fewer) anatomical regions (eg, the Collaborative Health Outcomes Information Registry [CHOIR] body template has 74 divisions to click [7]). In addition, assumptions must be made about where the anatomical locations of pain are clinically relevant or important.

Freehand pain drawings, using body templates with undemarcated or blank body outlines, are unbiased. Some body templates supporting freehand drawings use a grid with a predefined number of cells (eg, GeoPain app with a 3D body template uses a grid of 2026 cells [9] and Manchester Digital Pain Manikin uses a grid of 12,800 cells for its 2D body template [15]). The *grid-based* templates also allow the easy generation of pain frequency maps because each location (a cell) is binary in nature and is either selected or not selected. However, as body templates come in a variety of shapes and sizes [5], the aspect ratio (ratio of width to height) of the body regions also changes, and it is therefore not possible to standardize a grid (no *one-size-fits-all*). The optimal grid granularity (resolution) of each body template must be assessed. The granularity of the grid determines the end-user experience and accuracy of the captured data [15].

The *grid-free* body template, as the name suggests, overcomes the problem of grid creation and selection and provides an assumption-free basis for pain drawings that most resemble paper drawings. The number of clickable locations is of the order of thousands (depending on the pencil size). In addition, the participant may choose to draw repeatedly in the same location, similar to what happens when pen and paper are used. The precise capture of drawn areas, along with no predefined locations (cells), poses considerable challenges in generating pain frequency maps. The drawings on grid-free templates require intricate pixel-level analysis to generate a pain frequency map, and the tools for generating such maps are lacking or not freely available [16-18].

Objectives

In this paper, we describe a novel and unbiased algorithm developed specifically to generate a spatial pain frequency map from freehand pain drawings on a generic 2D whole-body template. We also assessed the performance of the algorithm's Python script and demonstrated its utility by generating pain frequency maps from pain drawings obtained from 2 clinical data sets.

Methods

Creating the Digital Body Template (Manikin)

An image outline (or template) of the human body is embedded in an HTML page to create a digital body template (manikin) to capture drawings of pain locations.

The responsiveness of the body template (ie, the ability to highlight or zoom in or out of a region) requires vectors, which are created using lines, points, and shapes to represent the different regions (demarcations) of the body. Using Scalable Vector Graphics (SVG), an XML-based language, it is possible to display vectors and create a responsive body template.

The locations of pain can be recorded on any body template by inserting basic SVG elements such as circles and rectangles. These shapes are inserted by specifying their position and size as the core attributes. For example, the circle requires coordinates (*cx*, *cy*) of the center and the radius (*r*), and the rectangle requires coordinates (*x*, *y*) of the top left corner along with *width* and *height*. In addition to the core attributes, the SVG elements can also contain style-related attributes (eg, fill, visibility, and opacity) and any number of customized attributes with prefix *data-* (eg, data-region and data-date-inserted).

XSL•FO
RenderX

The circle and rectangle elements can be created in SVG:

<circle cx="5" cy="5" r="10" fill="green" visibility="hidden" />

<rect x="5" y="5" width="20" height="10" fill="red" opacity="0.8" data-region="front" />

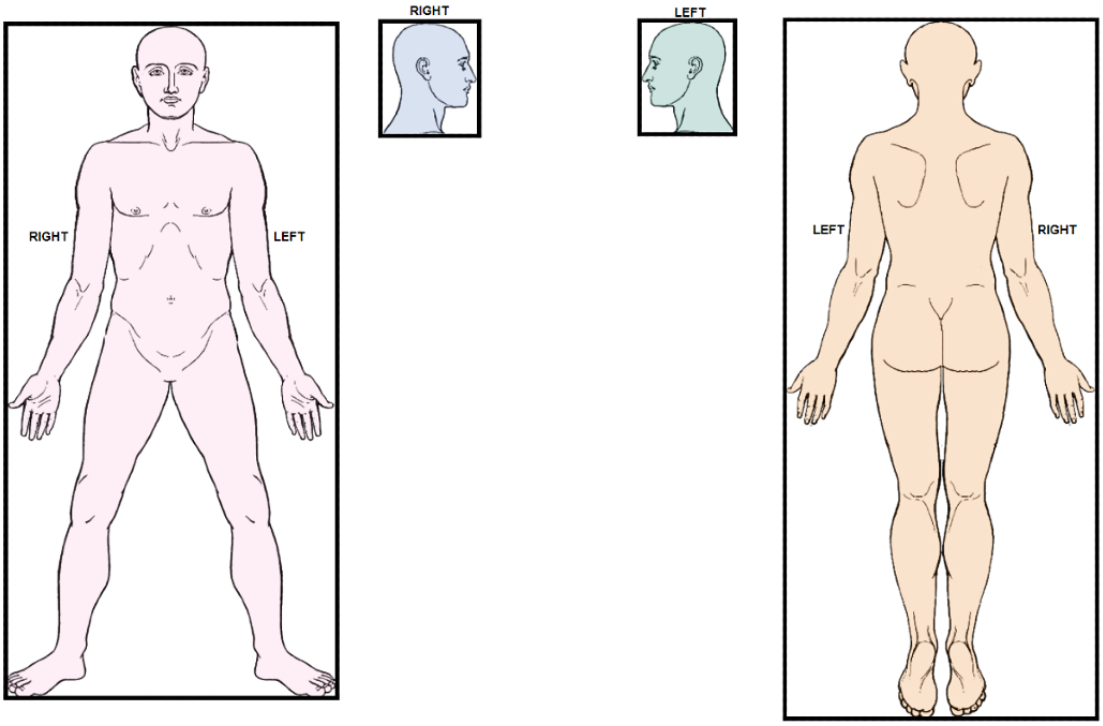
We used the SVG *<rect>* element to record the location of pain because the intersection of the 2 rectangles is always a rectangle (consistent geometry) that is required for the algorithm to function.

We downloaded a sexless human body outline image (TIFF), which is an adaptation of one of the oldest templates from the early works of Sir Henry Head [19]. The image has 4 views of the human body in 2D: coronal (front and back) and sagittal (left and right). The image was modified to demarcate 4 nonoverlapping regions: the front, side head (right), side head (left), and back. For making regions *clickable*, the body image was vectorized and converted into SVG using the Inkscape Editor (version 1.1) [20]. Vectorization produced 4 vectors (beziergons), where each vector has an associated bounding box, which is the tightest fitting rectangle that encloses all points on the vector (Figure 1).

In order to facilitate data capture from drawings of bodily pain locations, we first developed *Body Pain Map*, a tool using Linux-Apache-Perl-MongoDB–based infrastructure. The HTML5 webpage of the tool consists of a 2-column layout where the right column embeds the SVG body template and the left column contains 7 JavaScript-powered control elements (Figure 2), which are (1) size of rectangular pencil tip (*<select>* element with 3 options, small, medium, and large, signifying red squares of dimensions: 10, 30, and 60). The dimension of the smallest square was based on the smallest width found on the body template (eg, little finger), (2) zoom in (*<button>* element) to allow a closer view, (3) zoom out (*<button>* element) to allow a wider view, (4) Erase (*<button>* element) to remove a previous pain recording, (5) undo eraser (*<button>* element) to restate a previously removed recording, (6) clear (*<button>* element) to remove all recordings, and (7) submit (*<button>* element) to submit drawings for storage in the MongoDB database for analysis and reconstruction purposes. A blank body template can be submitted (eg, when the participant has no bodily pain to draw).

For making the Body Pain Map easily accessible as a web-based tool, we placed the HTML document on a secure Linux machine running the Secure Socket Layer (SSL)–enabled Apache HTTP web server (Figure 3). The tool can be accessed on the web [21] and tested on modern web browsers (eg, Google Chrome, Mozilla Firefox, and Microsoft Edge).

Figure 1. The body template (manikin) shows 4 regions. The tightest fitted rectangle (bounding box) enclosing each region is shown in black.



XSL•FO
RenderX

Figure 2. HTML webpage with a 2-column layout. The left column contains all control elements (buttons) and the right column contains the Scalable Vector Graphics body template. The participant can click anywhere on the template (divided into 4 regions) to locate their pain; a pain drawing of body regions, which comprises overlapping squares (rectangles of same width and height) of varying sizes is shown here. These drawings are input to the algorithm.

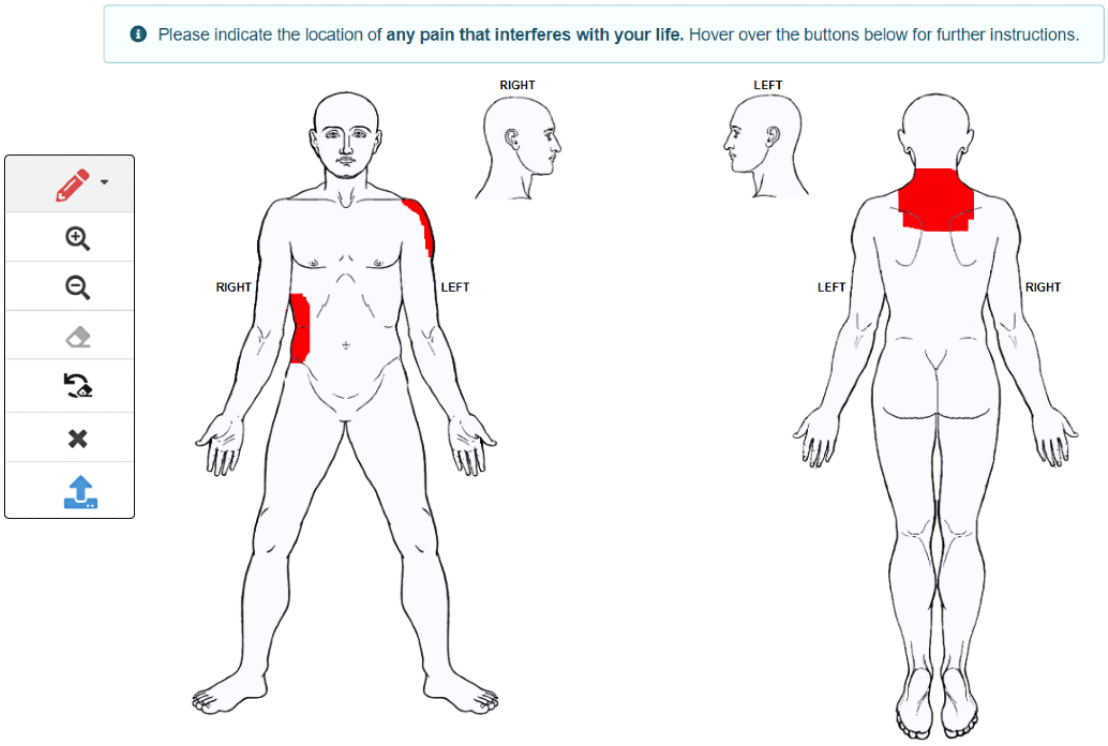
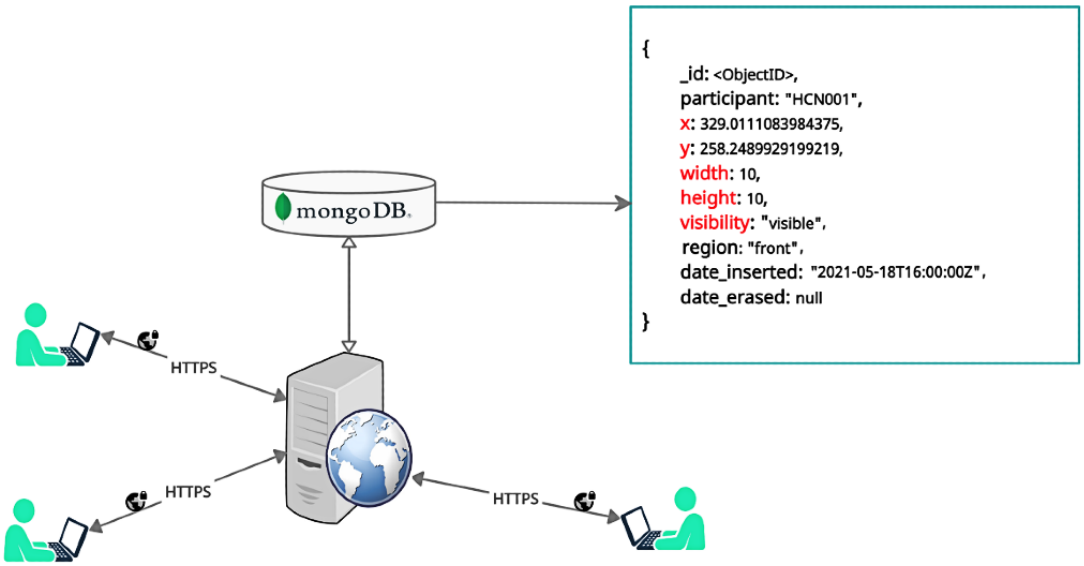


Figure 3. Each participant accesses the Body Pain Map tool through their web browser securely. The pain drawings are captured and stored in a MongoDB database. A sample JSON document of the pain recording (denoting Scalable Vector Graphics <rect> element) is shown as an example with core attributes in the color red.



Developing the Algorithm to Generate a Pain Frequency Map

Overview

The pain frequency map is generated by superimposing several drawings made by the participants using a rectangular pencil tip on a given body template. The drawing from each participant indicates where the pain is located on the body and can comprise multiple overlapping rectangles. The degree to which the rectangles overlap when all drawings are superimposed is of interest. The algorithm described in the *Overlap Computation Algorithm* section generates a pain frequency map that denotes the proportion of overlap in the rectangles between participant drawings. By default, the areas where the proportion of overlap is higher (ie, where pain is more commonly located) are redder in color. Areas with less overlap appear less red. Areas that are colored white are regions within the body template where no participant has drawn.

XSL•FO
RenderX

Overlap Computation Algorithm

The key steps in the overlap computation algorithm are (1) data decomposition, (2) creating partitions, (3) merging partitions to create nonoverlapping rectangles, and (4) optimizing nonoverlapping rectangles.

Data Decomposition

The first step toward 2D overlap computation is the decomposition of the source data from all participants' drawings on a given 2D body template.

Let *T* be the total number of participants, *P* (|P| ≤ T) be the set of strings denoting all participants with a drawing, *R* be the set of visible and unique rectangles from all participants in the *front* region of the body template, and *I* be the index set of set R, we assume that all participants in P contribute a drawing each, and as each participant drawing contains at least one rectangle, we define the relationship between rectangles (indexes) and participants as a surjective function *rectToParticipant*:

$$rectToParticipant : I \rightarrow P$$
$$\forall p \in P,\ \exists i \in I,\ rectToParticipant(i) = p$$

Assuming that each rectangle has attributes *x* (x coordinate of the top left corner), *y* (y coordinate of the top left corner), *w* (width) and *h* (height) we decompose rectangles along the x- and y-axis using the functions *coordX* and *coordY*:

$$Ex = \{(x, x+w) \mid x \in \mathbb{R} \wedge w \in \mathbb{R}^+\}$$
$$coordX : Ex \rightarrow Cx \subset \mathcal{P}(I)$$
$$coordX(x, x+w) = \{i \in I \mid \pi_X(r_i \mid r_i \in R) \cap \{(x, x+w)\} \neq \varnothing\}$$
$$Ey = \{(y, y+h) \mid y \in \mathbb{R} \wedge h \in \mathbb{R}^+\}$$
$$coordY : Ey \rightarrow Cy \subset \mathcal{P}(I)$$
$$coordY(y, y+h) = \{i \in I \mid \pi_Y(r_i \mid r_i \in R) \cap \{(y, y+h)\} \neq \varnothing\}$$

Creating Partitions

In this step, we create nonoverlapping partitions along the x- and y-axis. Each partition is an interval window of varying size and represents an area enclosed by at least one rectangle.

Let *E* be the domain and *C* (family of sets) be the range of the function *coordX* or *coordY* depending on the axis.

$$E = \{(e_i,\ e_{i+1})_{i=1} \mid i \in \mathbb{N}\}$$
$$C = \{(c_i)_{i=1} \mid i \in \mathbb{N}\} \subset \mathcal{P}(I)$$

The function *partition* for creating partitions along the x- and y-axis is defined as follows:

$$S = \{(s_k)_{k=1}^{n} | s_k \in \pi_1(E) \frown \pi_2(E)\}$$
$$P = \{(s_k, s_{k+1})\}$$

$$partition : P \rightarrow D \subset \mathcal{P}(I)$$
$$partition(s_k, s_{k+1}) = \bigcup_{i \in \mathbb{N}} c_i \mid (s_k, s_{k+1}) \subseteq (e_i, e_{i+1})$$

Merging Partitions to Create Nonoverlapping Rectangles

This step involves merging the X and Y partitions (nonoverlapping interval windows) to create nonoverlapping rectangles.

Let *Px* and *Dx* (family of sets) be the domain and range of *partitionX* and *Py* and *Dy* (family of sets) be the domain and range of *partition.*

$$Px = \{(px_i)_{i=1}^{m} \mid m \in \mathbb{N}\}$$
$$Dx = \{(dx_i)_{i=1}^{m} \mid m \in \mathbb{N}\} \subset \mathcal{P}(I)$$

$$Py = \{(py_j)_{j=1}^{n} \mid n \in \mathbb{N}\}$$
$$Dy = \{(dy_j)_{j=1}^{n} \mid n \in \mathbb{N}\} \subset \mathcal{P}(I)$$

We define *O* (a family of sets of participants *P*) as a set of all overlaps. The function *rectToParticipant* defined earlier is used to map the rectangles to their corresponding participants.

$$O = \{\{rectToParticipant(r) \mid r \in dx_i \cap dy_j \wedge r \neq \emptyset\}\} \subset \mathcal{P}(P)$$

For any *o* (*o* ∈ *O*) the set of nonoverlapping rectangles R_{ME} can then be defined as

$$R_{ME} = \{(px_i, py_j) \mid dx_i \cap dy_j = o \wedge i \in [1, n] \wedge j \in [1, m]\}$$

Each element in R_{ME} represents a unique, nonoverlapping location in the *front* region of the body template and carries an overlap frequency |o| or proportion given by $\frac{|o|}{T}$.

Optimizing Nonoverlapping Rectangles

In the previous step we generated nonoverlapping rectangles some of which might be adjoined.

This step describes the merging of adjoining rectangles for all the observed overlaps (*O*). The merging of rectangles is important for two main reasons: (1) to optimize the dimensions of the rectangles (ie, as wide or long as possible) and (2) to reduce the size of the output file (SVG) for efficient rendering by the web browser.

Let r_1 and r_2 be 2 nonoverlapping rectangles associated to the overlap *o* (*o* ∈ *O*).

$$r_1 = (x_1, y_1, w_1, h_1)$$
$$r_2 = (x_2, y_2, w_2, h_2)$$

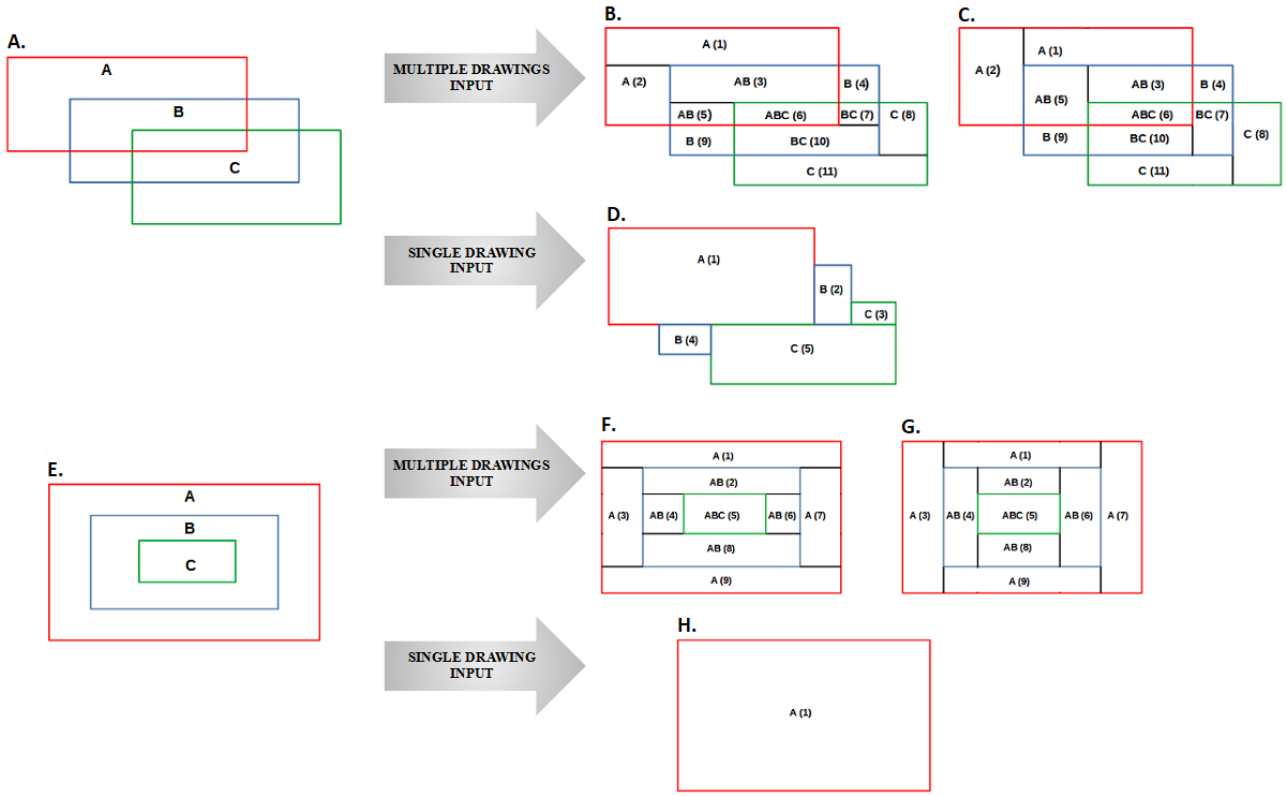
The 2 rectangles r_1 and r_2 are eligible for horizontal or vertical merging, provided that either of the following conditions is met.

$$r_{merged} = \begin{cases} (x_1, y_1, w_1 + w_2, h_1) & if\ x_1 + w_1 = x_2\ \ and\ \ h_1 = h_2 \\ (x_1, y_1, w_1, h_1 + h_2) & if\ y_1 + h_1 = y_2\ \ and\ \ w_1 = w_2 \end{cases}$$

Horizontal merging produced wider rectangles, whereas vertical merging produced longer rectangles (Figure 4).

XSL•FO
RenderX

Figure 4. Illustration of the algorithm's input and output. (A) A, B, and C are individuals who have drawn red, blue and green rectangles that overlap on a body template. (B) These overlapping rectangles are inputted to the algorithm. The output is 11 nonoverlapping rectangles (horizontal layout), named 1 to 11 in parentheses. Each rectangle represents a location with the proportion of pain. For example, A(1) = 1/3 , AB (3) = 2/3, ABC (6) = 3/3. (C) This shows the output from the algorithm in the vertical layout. The choice of layout (horizontal or wider rectangles or vertical or longer rectangles) can be made as per the locations of pain. For the legs, the rectangles might be better visualized when the layout is vertical, whereas horizontal layout is preferred for the abdominal area. (D) The illustration of the algorithm's output when the input is a single drawing made by one individual (A, B and C are the same individual). In this case, the output is simply 5 nonoverlapping rectangles instead of 11 each with a proportion of 1.0. (E) A, B and C are individuals who have drawn red, blue and green rectangles that overlap in a nested fashion, on a body template. These nested rectangles when inputted to the algorithm produce 9 nonoverlapping rectangles, with (F) showing the horizontal and (G) showing the vertical layout. (H) The demonstration of the output (ie, 1 nonoverlapping rectangle) when the input is a single drawing consisting of nested rectangles. This tends to occur when the individual elects to switch between pencil (tip) size: small, medium and large.



Algorithm Analysis Pipeline Construction

For implementing the overlap computation algorithm, we developed a command line workflow in Python programming language release 3.9 [22], which involves 3 steps (scripts) described further. The scripts can be run in either sequential or pipeline mode (using the named pipe command "|"). Each script performs a specific operation and accepts parameters in the form of command line options and arguments. The scripts use some core Python modules that are responsible for reading command line arguments, parsing and validating SVG to be processed, and producing the output for the subsequent step in the workflow. The following scripts can be obtained by contacting the corresponding author (AD):

1. *extract_data.py*: this script is only used if the pain drawings are originally saved as SVG files (a file per participant). The script extracts all the pain recordings denoted by *<rect>* elements with the same width and height attributes. The script produces a CSV output. The columns in the output are (1) *participant*: the identifier of the drawing, (2) *x*: the x coordinate of the top left corner of the rectangle, (3) *y*: the y coordinate of the top left corner of the rectangle, (3) *width*: the width of the rectangle, (4) *height*: the height of the rectangle (same as width), (5) *region*: the region of the body template where the recording belongs (eg, front or back), and (6) *visibility*: the visibility status of the recording (hidden means erased). The script also provides the option of including or excluding *empty* files. These files represent instances when the participant had no pain to draw or indicate on the body template. The script also allows users to extract data from specific body regions (eg, front or back).
2. *compute_overlap.py*: this script implements the overlap computation algorithm to produce nonoverlapping rectangles. Each rectangle represents a unique location on the body template. The script accepts input in CSV format with columns, namely, *participant*, *x*, *y*, *width*, *height*, and *region*. The script returns a CSV output where rows are nonoverlapping rectangles, and the columns (rectangles' attributes) are (1) *x*: the x coordinate of the top left corner of the rectangle, (2) *y*: the y coordinate of the top left corner of the rectangle, (3) *width*: the width of the rectangle, (4) *height*: the height of the rectangle, (5) *area*: the area of the rectangle, (6) *overlap*: the identifiers of drawings that overlap, (7) *overlap_frequency*: the number of identifiers that overlap, and (8) *overlap_proportion*: the proportion of identifiers that overlap. The proportion is calculated from the total number of drawings, which may include empty

XSL•FO
RenderX

drawings. The script also allows users to filter the output by providing the range for overlap (frequency) and thresholds for the width and height of the rectangles.

3. *plot_heatmap.py*: this script plots the nonoverlapping rectangles on the blank SVG body template and generates a pain frequency map (heat map) as an HTML file. The color of the pain frequency map can be specified either by its native name or as a hexadecimal color code (default #ff0000 or red). The intensity (shade or gradient) of the participant overlap on the pain frequency map is displayed using the opacity attribute of an SVG element. Opacity is any number strictly between 0 and 1. For optimum color coding, the opacity for each output rectangle is calculated as follows:

$$opacity = \frac{overlap_frequency}{overlap_frequency_{max}}$$

Data Generation for Algorithm Implementation

Simulation

We examined the performance of the algorithm's Python script *compute_overlap.py* in 2 separate simulations on a machine (Intel Xeon[R] Silver 4110 CPU@2.10GHz and 16-GB RAM) running Ubuntu 18.04. Given an XY plane of dimensions 1000×1000 with the origin at (0,0), let *X* and *Y* be the set of all natural numbers on the x- and y-axis. The set of all ordered pairs *P* is denoted by the Cartesian product $X \times Y$ of sets *X*, *Y*.

Simulation 1

We assumed that the drawings from every individual consisted of a single rectangle (analogous to a single mouse *click* on the body template). This situation is extremely unlikely; however, the purpose of this simulation was to test the ability of the algorithm to compute an overlap, given a set of overlapping rectangles.

A total of 10 data sets were generated for this simulation. The first data set consisted of 10,000 rectangles, and for each consecutive data set, the number of rectangles was increased by 10,000. For each data set, the coordinates of the origin of the rectangles were sampled without replacement from the set of ordered pairs (*P*), and the dimensions (width and height) were sampled with replacement from a sequence starting at 10 and ending at 100 (incremental step is 10).

Simulation 2

We assumed that a typical participant drawing of pain locations consisted of 100 rectangles (equivalent to 100 mouse *clicks* on the body template). For this simulation, we sought to assess performance with an increasing number of participants.

A total of 10 data sets were generated for this simulation. The first data set consisted of 100 participants, and for each consecutive data set, we increased the number of participants by 100. For each participant, the origin of the rectangles was sampled without replacement from the set of ordered pairs (*P*), and the dimensions (width and height) were sampled with replacement from a sequence starting at 10 and ending at 100 (incremental step is 10).

Each simulation produced 10 CSV files for analysis by the algorithm. The CSV file consisted of the columns, namely, *participant*, *x*, *y*, *width*, *height*, and *region*. For both simulations, there was only one region (the XY plane); therefore, the *region* was simply labeled *xy-plane*.

Actual

Finally, we used the digital pain drawings from 2 clinical data sets; the first data set (data set 1) consisted of 23 individuals who were screened for a clinical drug trial (ISRCTN68734605) [23], and the second data set (data set 2) comprised 30 participants with chronic back pain who were recruited into a separate study [24]. The pain drawings were stored as MongoDB documents (Figure 3), which were organized into subdocuments ordered by the date and time of insertion. The virtual pencil tip used for the drawings was a square, an SVG <*rect*> element, where the width and height attributes were the same (Figure 2).

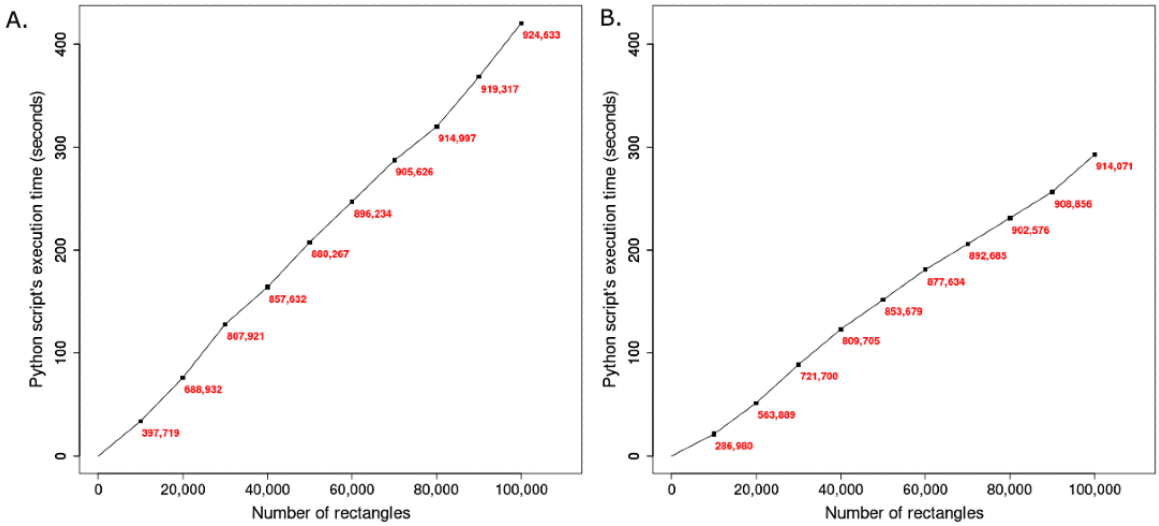
Results

Simulation

The simulations were carried out to assess the performance of the algorithm's Python script *compute_overlap.py*. The simulations reveal that the script's execution time (seconds) increases linearly as a function of the number of input rectangles (Figure 5), up to 400 seconds for 100,000 rectangles (simulation 1). Assuming that a typical participant drawing consists of 100 rectangles, the execution time for 1000 participant drawings is 300 seconds (simulation 2).

XSL•FO
RenderX

Figure 5. Simulations to assess the performance of the algorithm's Python script – compute_overlap.py. The y-axis shows the execution time of the script in seconds and the x-axis shows the number of rectangles handled by the script. The output from the script, the number of nonoverlapping rectangles is shown in red. The plot (A) shows the script's performance on the data sets from simulation 1 where the number of rectangles per participant was set to 1, and plot (B) shows the performance on the data sets generated in simulation 2 where the number of rectangles per participant drawing was set to 100.



Actual

We first extracted all drawings (<rect> elements with the same width and height attributes) from the 2 data sets stored in the MongoDB database into CSV files. Data set 1 produced 4016 and data set 2 produced 4167 recordings, respectively, sorted by the participant identifier and the date and time of insertion. The characteristics of the participants from the 2 data sets pertaining to the *Body Pain Map* exercise are summarized in Table 1.

For generating pain frequency maps, we only used visible and unique (nonidentical) recordings from the 2 clinical data sets. Identical recordings from the same participant (<*rect*> elements with complete overlay) were filtered based on the recording's last visibility status. If visibility was hidden (meaning erased), the recording was excluded. This exercise produced 2 CSV files (one file per data set) with 3242 and 3993 rows, respectively, signifying unique and visible recordings across all 4 regions, namely, front, side (right), side (left), and back as summarized in Table 2.

The CSV data sets were independently processed by the Python script *compute_overlap.py* on a machine (Intel(R) Xeon (R) Silver 4110 CPU @ 2.10GHz and 16 GB RAM) running Ubuntu 18.04. The script processed the first CSV data set in 4 seconds and produced 6653 nonoverlapping rectangles with an overlap frequency between 1 and 8. The second CSV data set was processed in 7 seconds and produced 6010 nonoverlapping rectangles with an overlap frequency ranging from 1 to 21. Each nonoverlapping rectangle represents a unique location on the 2D body template and carries an overlap frequency, which denotes the number of participants with pain at that location.

The nonoverlapping rectangles for the 2 clinical data sets were subsequently plotted on the body template using the Python script *plot_heatmap.py*, and a pain frequency map was generated for data set 1 (Figure 6) and data set 2 (Figure 7).

Table 1. Pencil and eraser clicks made by the participants from the 2 clinical data sets while using the Body Pain Map tool.

Characteristic	Data set 1	Data set 2
Participants, N	23	30
Clicks, mean (SD)		
Pencil[a]	174.6 (208.5)	138.9 (108.2)
Eraser[b]	16.7 (32.7)	1.5 (4.2)
Visible and unique pencil clicks, mean (SD)	141.0 (176.5)	133.1 (105.5)
Mean pencil size[c], mean (SD)	26.6 (10.3)	25.2 (6.0)

[a]Each pencil click denotes a recording of the pain location and creates a Scalable Vector Graphics <*rect*> element with the same width and height attributes (a square).

[b]The eraser click hides the previously inserted Scalable Vector Graphics <*rect*> element.

[c]On the basis of the visible and unique pencil clicks to record (draw) pain locations.

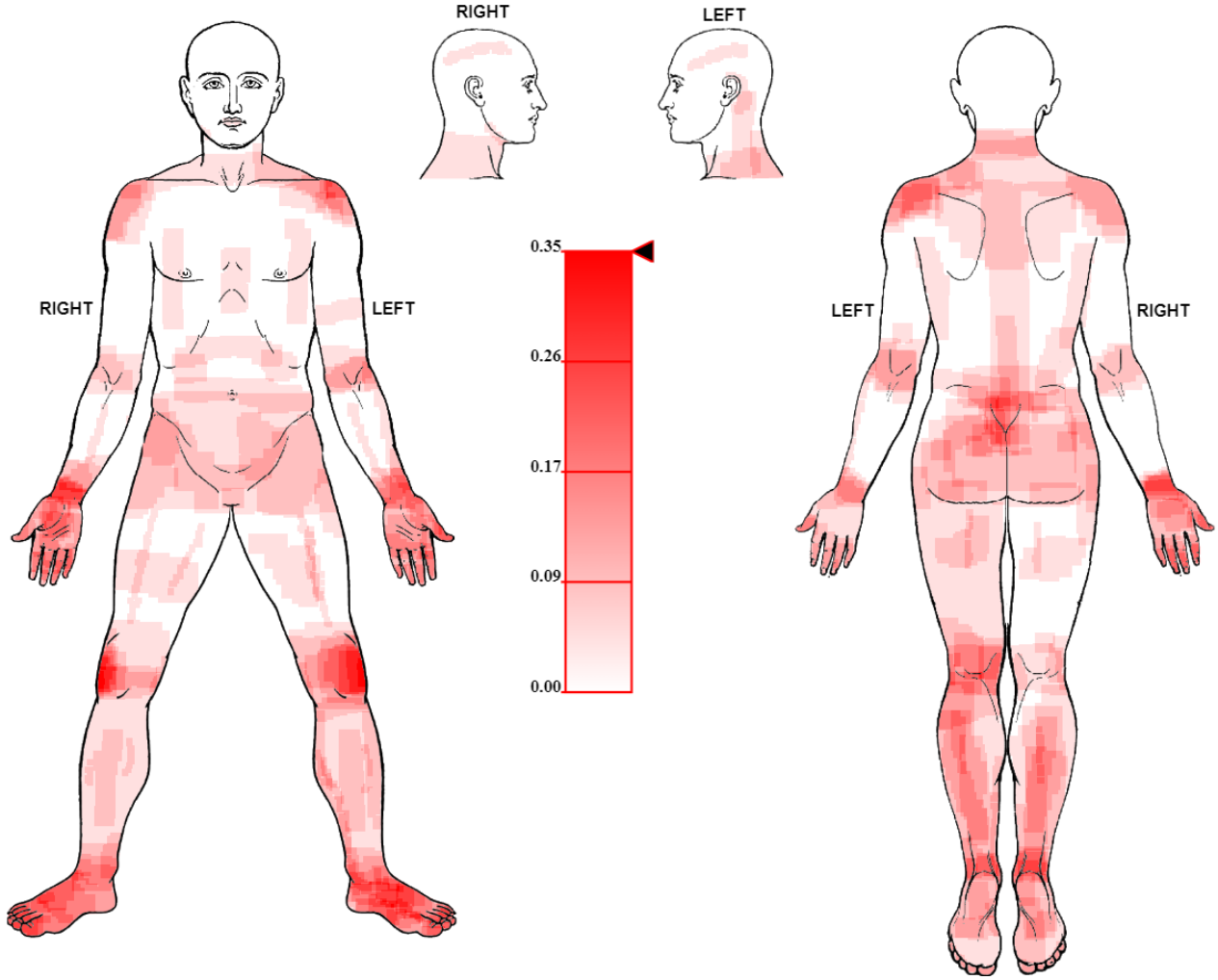
XSL•FO
RenderX

Table 2. Visible and unique pencil recordings made in each region for 2 clinical data sets[a].

Characteristic	Data set 1	Data set 2
Recordings, N	3242	3993
Region, n (%)		
Front	1381 (42.6)	800 (20.0)
Side (right)	80 (2.5)	0 (0.0)
Side (left)	103 (3.2)	11 (0.3)
Back	1678 (51.7)	3182 (79.7)

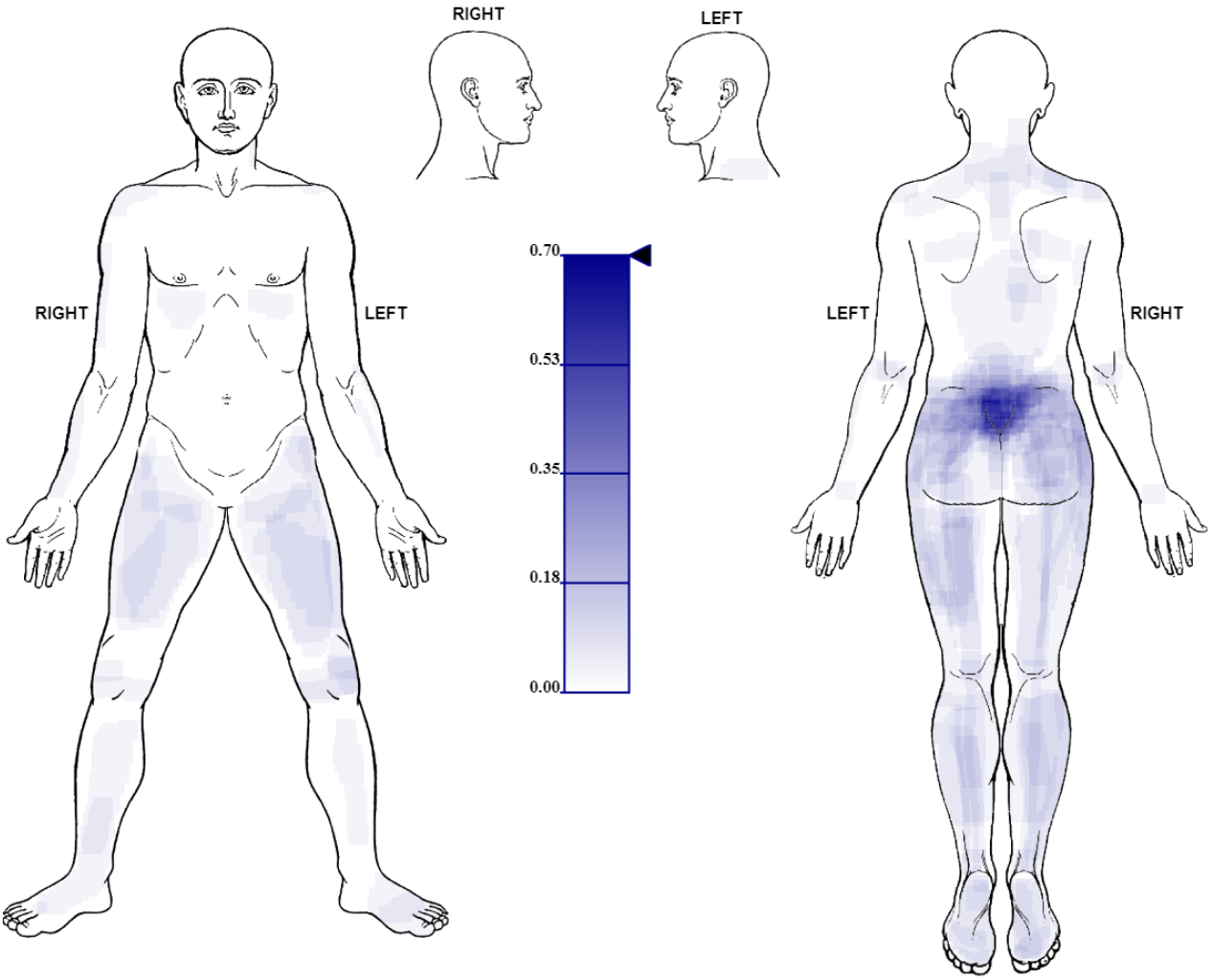
[a]Data are derived from the Scalable Vector Graphics drawings and provided to the algorithm's Python scripts to generate a pain frequency map.

Figure 6. Illustration of the pain frequency map produced from freehand pain drawings (data set 1) obtained from patients (N=23) who were screened for a clinical drug trial. In the interactive map (HTML format), the user can slide the black pointer on the gradient bar (shown in the middle) to view locations based on the overlap threshold (ie, <= overlap proportion) and save the map as a PNG file.



XSL•FO
RenderX

Figure 7. Illustration of the pain frequency map produced from freehand pain drawings (data set 2) obtained from patients (N=30) diagnosed with chronic primary back pain.



Discussion

Principal Findings

Drawing pain locations on body templates is widely and increasingly being used in research and pain clinics. These are frequently part of questionnaires (eg, the Brief Pain Inventory [25]). The topography of bodily pain is often summarized as a pain frequency map but relies on digitization of paper drawings, which is labor intensive and may be infeasible for larger studies [26]. Although freehand digital drawing tools are available to capture pain locations, they are often restricted to specific body templates [6]. Other tools parcellate the body (eg, the CHOIR Body Map [7] and Michigan Body Map [13]), which allows easy generation of pain frequency maps but lacks the resolution required.

In this paper, we first described the Body Pain Map [21], a web-based platform used to capture freehand pain drawings as SVG *<rect>* elements and then thoroughly described the algorithm to generate a pain frequency map from the drawings. We further described the implementation of the algorithm as a Python-based analysis pipeline and tested its performance in 2 different simulations. We demonstrated the utility of the algorithm by producing pain frequency maps (Figure 6 and Figure 7) from freehand drawings obtained from 23 patients screened for a clinical drug trial (ISRCTN68734605) [23] and 30 patients diagnosed with chronic primary back pain [24].

The key advantage of this algorithm is that it can handle data from any undemarcated body template. In other words, the body template does not need to be divided using a mesh with a fixed number of cells (eg, GeoPain, a body surface map rendered on a 3D manikin [9] and Manchester Digital Pain Manikin [15]) or demarcated into regions with anatomical labels (eg, CHOIR [7]). Only the outline or perimeter of the body template requires demarcation. The label-free and grid-free approach is unbiased and allows the participants to freely draw the locations of their bodily pain, closely mimicking the experience of drawing on paper. Although the user is unlikely to draw in the entire space, given the nature of freehand drawing, they can draw repeatedly in the same area. This results in redundant data, which are not usually of interest to most clinician researchers. Our algorithm processes raw (source pixel) data in grid-free space to address this difficulty, and in doing so, it also achieves lossless compression.

The algorithm accepts a single CSV (plain text) file with pain recordings captured as rectangles and produces nonoverlapping rectangles. Each nonoverlapping rectangle is a unique location

XSL•FO
RenderX

on the body template and carries an overlap frequency, which denotes the number of participants with pain in that location. The output, when transformed into an HTML file, can be feasibly rendered on modern web browsers (eg, Google Chrome, Mozilla Firefox, or Microsoft Edge) or printed at the desired resolution for publishing as we have shown for 2 clinical data sets. The algorithm also contains features to optimize the display of the pain frequency map. Rectangles can be produced either in a horizontal (wider) or vertical layout (longer; Figure 4). For example, for the leg, the rectangles may be better visualized with a vertical layout, whereas a horizontal layout is preferred for wider regions such as the abdomen.

As the individual can draw freely on the body template, the output (ie, nonoverlapping rectangles) produced by the algorithm is highly granular. Given the required spatial information (x-y coordinates within the body region), it is possible to retrospectively create a grid and reassign the cells and anatomical labels as required. The output from the algorithm can be subjected to coordinate transformation using linear algebra and labeled to allow harmonization with other body atlases or templates with anatomical labels (eg, arms and legs) in nature [7]. The unbiased output can also be trained to provide demarcations for a given pain disorder within any 2D body template. The demarcations or boundaries of areas for an anatomical label (eg, shoulder) may also be determined empirically through feedback from any group of individuals. It is also possible to acquire drawings from gender-specific body templates provided by the Hannover Medical School [8] and subsequently use the coordinate transformation to harmonize the algorithm's output on a gender-neutral template for comparisons between sexes.

Limitations

The algorithm generates nonoverlapping rectangles without any knowledge of the region boundaries within the 2D body template. Hence, some rectangles in the output may fall partly outside these regions. This overflow occurs when drawings include the border (or are close to the border) of the body regions. For generating pain frequency maps, this is not a problem because rectangles falling outside the regions can be easily masked (hidden) using SVG. For other analysis purposes, the problem can be feasibly mitigated by approaches such as (1) dividing the body template into small regions and applying bounding box correction to filter output nonoverlapping rectangles by region or (2) creating a *null* drawing that delicately fills the entire body template, ensuring that the <*rect*> elements are within the regional boundaries. With this approach, the *null* drawing is added to actual participant drawings, and the algorithm is executed on the combined data set. In the resulting output, only the nonoverlapping rectangles that are common to both the *null* drawing and the actual drawings are retained, and the rest (overlap frequency is 1) are discarded.

The algorithm is optimized to generate pain frequency data only for the locations of the body template on which at least one participant has drawn. Adding the *null* drawing would allow representation (in the pain frequency map) of body locations within the template that no participant has drawn and may also be required for inferential statistics. As the *null* drawing is specific to a body template, it does not have to be recreated, and the approach can be applied to prospective data sets (drawings), provided they are acquired using the same body template.

Comparison With Prior Work

Software tools (R package, CHOIR Body Map [27]) have recently been developed to generate co-occurrence maps. The map shows the number of times 2 locations on the body template are endorsed together by patients in a data set. However, these tools are only applicable to the CHOIR body template [7], where the participant can only click on 74 predefined locations.

Generating a pain frequency map for body templates with either demarcations or grid is straightforward because the locations are predefined and fixed, and it is a case of deducing the number of participants that selected each location [14].

Studies in which a nondemarcated and grid-free body template was used for freehand drawings used customized tools for the simultaneous analysis of their drawings. These tools are not freely available in the public domain [16-18]. Other studies [28,29] converted their drawings (originally saved as PNG images) into NIfTI format [30] and analyzed them using image-processing tools provided by the Functional Magnetic Resonance Imaging of the Brain (FMRIB) Software Library [31].

The primary purpose of our algorithm and other methods used in previous studies is to generate pain frequency maps, which requires simultaneous analysis of all freehand drawings provided by participants. Previous studies have stored freehand pain drawings as bitmap images (eg, PNG); therefore, the generation of a pain frequency map requires the extraction and analysis of all pixels [16-18,28,29].

Our algorithm processes drawings in which pain locations are indicated using rectangles (the pencil tip is a square, which is fundamentally a rectangle). This has several advantages, such as the fact that the input from the drawings can be simply stored as a CSV (plain text) file because all locations are represented as rectangles with attributes *x*, *y*, *width*, and *height*. This is also more efficient than the storage and extraction of pixels from the drawings. The output also comprises rectangles and can be stored as CSV files for statistical analyses and visualized using images (eg, SVG and PNG). We used SVG because it allows the reconstruction and visualization of the input and output rectangles at the desired resolution. CSV storage also facilitates the merging of several independent data sets acquired using the same body template for a combined analysis.

Conclusions

Body maps have long been used in research and clinical practice to facilitate communication between pain and other sensations [32]. The choice of a drawing tool or body part selection depends on the nature of the research question and the participants. It is important to clinically validate any digital tool used to capture the topography of body sensations [33]. Our algorithm is primarily developed to render pain frequency maps for efficient display and printing, but the output (ie, nonoverlapping rectangles) can be readily subjected to statistical

XSL•FO
RenderX

analyses (eg, statistical comparisons of pain frequency maps between different patient cohorts or the same patient cohort over multiple time points). Although we chose to digitize and use a specific body template in our study, the algorithm described can process any number of freehand drawings on any 2D body template to produce a pain frequency map. The nonoverlapping rectangles generated by the algorithm can be labeled anatomically or mapped onto a grid to facilitate analyses and harmonization with other body templates.

Our freehand pain drawing tool (Body Pain Map [21]) uses resolution-independent and XML document object model–based SVG technology. However, our algorithm can also generate pain frequency maps from drawings created using other technologies (eg, HTML5 <*canvas*> element), provided that the pain locations are captured as rectangles and the location attributes (ie, *x*, *y*, *width*, and *height*) are accessible.

As the algorithm has already been implemented as a Python command line workflow, it is possible to schedule an automated pain frequency map construction through the cron daemon (Linux environment) and filter and visualize the output using several criteria (eg, region, overlap frequency, width, and height of rectangles). The Python scripts can be obtained by contacting the corresponding author (AD).

We envisage the use of the algorithm in clinical or research settings to facilitate fine-grain comparisons of human pain anatomy between disease diagnosis or disorders or as an outcome metric to guide the monitoring or discovery of treatments.

Acknowledgments

The authors would like to thank Ingrid Scholtes and Gill Ison for their help in obtaining the digital pain drawings used in the 2 data sets and Alasdair Kerslake for testing the Body Pain Map tool. They would also like to acknowledge Folkert Tangerman for his feedback regarding the algorithm. AD is presently supported by the UK Medical Research Council and Versus Arthritis Grant (MR/W002426/1) and Advanced Pain Discovery platform. Clinical data sets (data sets 1 and 2) were acquired from separate projects funded by the UK Medical Research Council (MR/J013129/1) and Versus Arthritis (grant 21538).

Conflicts of Interest

None declared.

References

1. Woolf CJ. What is this thing called pain? J Clin Invest 2010 Nov;120(11):3742-3744 [FREE Full text] [doi: 10.1172/JCI45178] [Medline: 21041955]
2. Poquet N, Lin C. The brief pain inventory (BPI). J Physiother 2016 Jan;62(1):52 [FREE Full text] [doi: 10.1016/j.jphys.2015.07.001] [Medline: 26303366]
3. Palmer H. Pain charts; a description of a technique whereby functional pain may be diagnosed from organic pain. N Z Med J 1949 Apr;48(264):187-213. [Medline: 18138692]
4. Jang A, MacLean D, Heer J. BodyDiagrams: improving communication of pain symptoms through drawing. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems. 2014 Presented at: CHI '14: CHI Conference on Human Factors in Computing Systems; Apr 26 - May 1, 2014; Toronto Ontario Canada. [doi: 10.1145/2556288.2557223]
5. Shaballout N, Neubert T, Boudreau S, Beissner F. From paper to digital applications of the pain drawing: systematic review of methodological milestones. JMIR Mhealth Uhealth 2019 Sep 05;7(9):e14569 [FREE Full text] [doi: 10.2196/14569] [Medline: 31489841]
6. Ali SM, Lau WJ, McBeth J, Dixon WG, van der Veer SN. Digital manikins to self-report pain on a smartphone: a systematic review of mobile apps. Eur J Pain 2021 Feb;25(2):327-338 [FREE Full text] [doi: 10.1002/ejp.1688] [Medline: 33113241]
7. Scherrer KH, Ziadni MS, Kong J, Sturgeon JA, Salmasi V, Hong J, et al. Development and validation of the collaborative health outcomes information registry body map. Pain Rep 2021;6(1):e880 [FREE Full text] [doi: 10.1097/PR9.0000000000000880] [Medline: 33490848]
8. Hannover body template. Figshare. URL: https://figshare.com/articles/dataset/Hannover_Body_Template/7637387/9 [accessed 2022-04-26]
9. Kaciroti N, DosSantos MF, Moura B, Bellile EL, Nascimento TD, Maslowski E, et al. Sensory-discriminative three-dimensional body pain mobile app measures versus traditional pain measurement with a visual analog scale: validation study. JMIR Mhealth Uhealth 2020 Aug 19;8(8):e17754 [FREE Full text] [doi: 10.2196/17754] [Medline: 32124732]
10. van den Ameele J, Fuge J, Pitceathly RD, Berry S, McIntyre Z, Hanna MG, et al. Chronic pain is common in mitochondrial disease. Neuromuscul Disord 2020 May;30(5):413-419 [FREE Full text] [doi: 10.1016/j.nmd.2020.02.017] [Medline: 32334903]
11. El-Damanawi R, Lee M, Harris T, Cowley LB, Scholtes I, Bond S, et al. Developing a patient-centred tool for pain measurement and evaluation in autosomal dominant polycystic kidney disease. Clin Kidney J 2021 Nov;14(11):2338-2348 [FREE Full text] [doi: 10.1093/ckj/sfaa259] [Medline: 34754429]
12. Nicholas M, Vlaeyen JW, Rief W, Barke A, Aziz Q, Benoliel R, IASP Taskforce for the Classification of Chronic Pain. The IASP classification of chronic pain for ICD-11: chronic primary pain. Pain 2019 Jan;160(1):28-37. [doi: 10.1097/j.pain.0000000000001390] [Medline: 30586068]

XSL•FO
RenderX

13. Brummett C, Bakshi R, Goesling J, Leung D, Moser S, Zollars J, et al. Preliminary validation of the Michigan Body Map. Pain 2016 Jun;157(6):1205-1212 [FREE Full text] [doi: 10.1097/j.pain.0000000000000506] [Medline: 26835782]
14. Alter BJ, Anderson NP, Gillman AG, Yin Q, Jeong J, Wasan AD. Hierarchical clustering by patient-reported pain distribution alone identifies distinct chronic pain subgroups differing by pain intensity, quality, and clinical outcomes. PLoS One 2021 Aug 4;16(8):e0254862 [FREE Full text] [doi: 10.1371/journal.pone.0254862] [Medline: 34347793]
15. van der Veer SN, Beukenhorst AL, Ali SM, James B, Silva P, McBeth J, et al. Development of a mobile digital manikin to measure pain location and intensity. Stud Health Technol Inform 2020 Jun 16;270:946-950. [doi: 10.3233/SHTI200301] [Medline: 32570521]
16. Rosner J, Lütolf R, Hostettler P, Villiger M, Clijsen R, Hohenauer E, et al. Assessment of neuropathic pain after spinal cord injury using quantitative pain drawings. Spinal Cord 2021 May 16;59(5):529-537 [FREE Full text] [doi: 10.1038/s41393-021-00616-6] [Medline: 33594250]
17. Cruder C, Falla D, Mangili F, Azzimonti L, Araújo LS, Williamon A, et al. Profiling the location and extent of musicians' pain using digital pain drawings. Pain Pract 2018 Jan;18(1):53-66 [FREE Full text] [doi: 10.1111/papr.12581] [Medline: 28466572]
18. Barbero M, Moresi F, Leoni D, Gatti R, Egloff M, Falla D. Test-retest reliability of pain extent and pain location using a novel method for pain drawing analysis. Eur J Pain 2015 Sep;19(8):1129-1138. [doi: 10.1002/ejp.636] [Medline: 25565607]
19. Beissner F, Brünner F, Fink M, Meissner K, Kaptchuk TJ, Napadow V. Placebo-induced somatic sensations: a multi-modal study of three different placebo interventions. PLoS One 2015 Apr 22;10(4):e0124808. [doi: 10.1371/journal.pone.0124808] [Medline: 25901350]
20. Inkscape 1.2. Inkscape. URL: https://inkscape.org/ [accessed 2022-04-26]
21. Body pain map. University of Cambridge. URL: https://pain.medschl.cam.ac.uk/body-pain-map/preview/61c797000aff0789b65d01e6 [accessed 2022-04-26]
22. Python 3.9.0. Python. URL: https://www.python.org/downloads/release/python-390/ [accessed 2022-04-26]
23. Bernard Healey SA, Scholtes I, Abrahams M, McNaughton PA, Menon DK, Lee MC. Role of hyperpolarization-activated cyclic nucleotide-gated ion channels in neuropathic pain: a proof-of-concept study of ivabradine in patients with chronic peripheral neuropathic pain. Pain Rep 2021 Nov;6(4):e967 [FREE Full text] [doi: 10.1097/PR9.0000000000000967] [Medline: 34712888]
24. RS Fibro v1.0. NHS Health Research Authority. URL: https://www.hra.nhs.uk/planning-and-improving-research/application-summaries/research-summaries/rs-fibro-v10/ [accessed 2022-05-03]
25. The brief pain inventory user guide. BPI User Guide. URL: https://www.mdanderson.org/documents/Departments-and-Divisions/Symptom-Research/BPI_UserGuide.pdf [accessed 2022-04-26]
26. Caseiro M, Woznowski-Vu A, De Oliveira AS, Reis FJ, Wideman TH. From paper to digitalized body map: a reliability study of the pain area. Pain Pract 2019 Jul;19(6):602-608. [doi: 10.1111/papr.12780] [Medline: 30884135]
27. emcramer / CHOIRBM. GitHub. URL: https://github.com/emcramer/CHOIRBM [accessed 2022-04-27]
28. Neubert T, Dusch M, Karst M, Beissner F. Designing a tablet-based software app for mapping bodily symptoms: usability evaluation and reproducibility analysis. JMIR Mhealth Uhealth 2018 May 30;6(5):e127. [doi: 10.2196/mhealth.8409] [Medline: 29848470]
29. Ellingsen D, Beissner F, Moher Alsady T, Lazaridou A, Paschali M, Berry M, et al. A picture is worth a thousand words: linking fibromyalgia pain widespreadness from digital pain drawings with pain catastrophizing and brain cross-network connectivity. Pain 2021 May 01;162(5):1352-1363 [FREE Full text] [doi: 10.1097/j.pain.0000000000002134] [Medline: 33230008]
30. NIfTI: Neuroimaging Informatics Technology Initiative. National Institute of Health. URL: https://nifti.nimh.nih.gov/ [accessed 2022-05-08]
31. FMRIB Software Library v6.0. FSL. URL: https://fsl.fmrib.ox.ac.uk/fsl/fslwiki [accessed 2022-04-26]
32. Schott GD. The cartography of pain: the evolving contribution of pain maps. Eur J Pain 2010 Sep;14(8):784-791. [doi: 10.1016/j.ejpain.2009.12.005] [Medline: 20079670]
33. Body mapping in research. In: Handbook of Research Methods in Health Social Sciences. Singapore: Springer; 2019.

Abbreviations

CHOIR: Collaborative Health Outcomes Information Registry
FMRIB: Functional Magnetic Resonance Imaging of the Brain
SSL: Secure Socket Layer
SVG: Scalable Vector Graphics

XSL•FO
RenderX

Edited by A Mavragani; submitted 20.01.22; peer-reviewed by F Beissner, D Wong; comments to author 27.04.22; revised version received 12.05.22; accepted 16.05.22; published 24.06.22

Please cite as:
Dixit A, Lee M
Quantification of Digital Body Maps for Pain: Development and Application of an Algorithm for Generating Pain Frequency Maps
JMIR Form Res 2022;6(6):e36687
URL: https://formative.jmir.org/2022/6/e36687
doi: 10.2196/36687
PMID: 35749160

©Abhishek Dixit, Michael Lee. Originally published in JMIR Formative Research (https://formative.jmir.org), 24.06.2022. This is an open-access article distributed under the terms of the Creative Commons Attribution License (https://creativecommons.org/licenses/by/4.0/), which permits unrestricted use, distribution, and reproduction in any medium, provided the original work, first published in JMIR Formative Research, is properly cited. The complete bibliographic information, a link to the original publication on https://formative.jmir.org, as well as this copyright and license information must be included.

XSL•FO
RenderX