UCLouvain

LSINC1114 – Analysis of biological data

Session 8 – Medical imaging and DICOM

Sébastien Jodogne
EPL / ICTEAM
UCLouvain, Belgium

Academic year 2022-2023

Outline

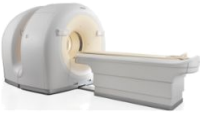
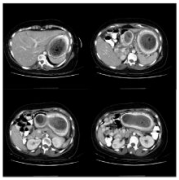
UCLouvain

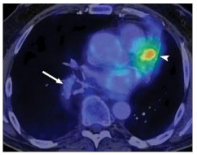
1. Brief overview of medical imaging
2. DICOM
3. HTML5 canvas

Brief overview of medical imaging

CT + MRI + PET-CT





- Belgium (2013): 33 millions of imaging studies for 11 millions of persons.
- Reason: Multimodal and longitudinal imaging (oncology, cardiovascular diseases, surgery, neurology...)

- Many lesions/disorders are internal → How can we detect them and select the best treatment?
- How can we see inside the body in a non-invasive way, i.e. without "opening" it with a scalpel?

Invasive

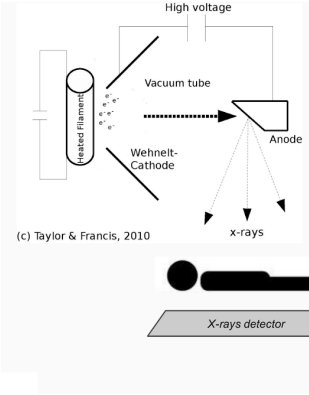
Non-invasive

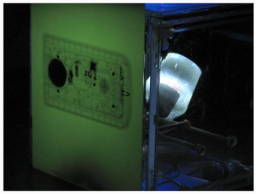
Credits: B. Macq, J. Lee, F. Peeters, G. Kerckhofs (LGBIO2050 @ UCLouvain)

It all began in 1895 with Wilhelm Conrad Röntgen's picture of his wife Bertha's hand:

Credits: B. Macq, J. Lee, F. Peeters, G. Kerckhofs (LGBIO2050 @ UCLouvain)

- An homogeneous beam of X-rays is produced by an X-ray generator and is directed towards an object.
- According to the density and composition of the different components of the object, X-rays are more or less absorbed.
- X-rays are then captured behind the object by a detector, which gives a 2D image as if it was transparent somehow.



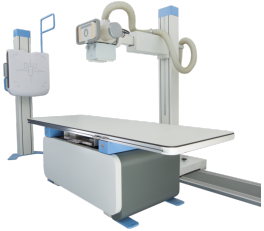


Credits: B. Macq, J. Lee, F. Peeters, G. Kerckhofs (LGBIO2050 @ UCLouvain)

- Conventional X-ray photographic plate, to be chemically developed, then scanned (RadioGraphic imaging = RG).
- Fluorescent screen, recorded by a visible-light camera (Radio Fluoroscopy = RF).
- Better: Photo-stimulable phosphor plate, where excited electrons are read using a laser, which is reusable (Computed Radiography = CR).
- Semiconductor detectors can directly create an image, since 2000's (Digital Radiography = DR).

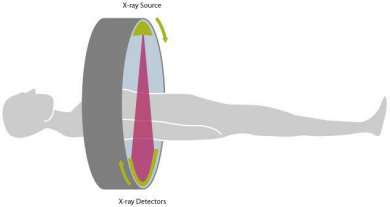
RF

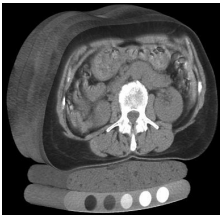
CR



DR

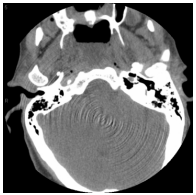
Radiography device rotating around the patient
$\rightarrow$ Tomographic reconstruction (inverse Radon transform, 1917) $\rightarrow$ 3D image





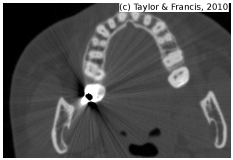
1971: EMI scanner
(Sir Godfrey Hounsfield)

https://thelongandshort.org/enterprise/how-the-beatles-revolutionised-medical-imaging

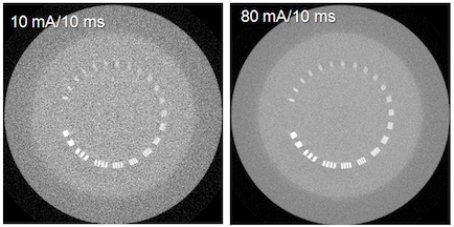


Ring artifact

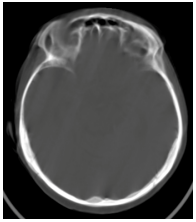
Ring artifact



Streak artifact (metal)



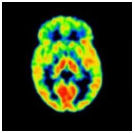
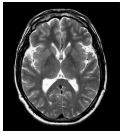
Noise / partial volume effect (low-dose)



Motion artifact

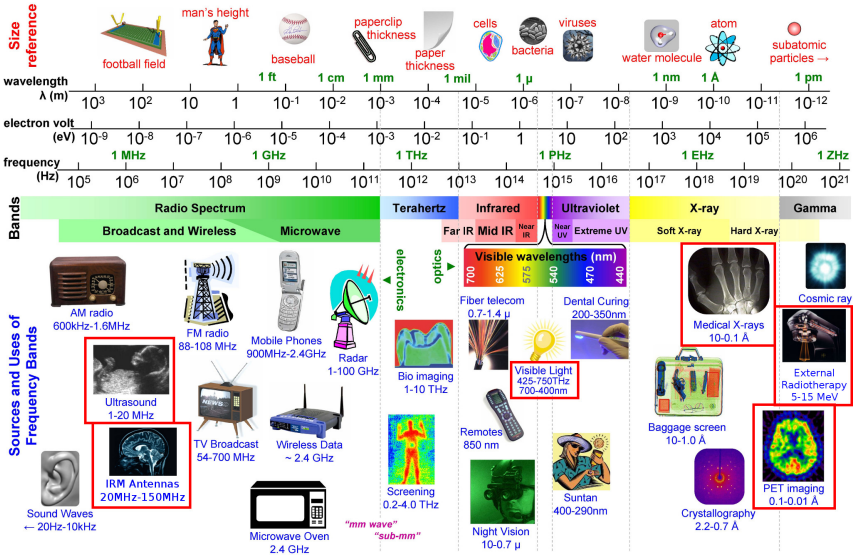
- Other waves and/or techniques are used in medical imaging:
  - DR - Digital radiography → X-ray transmission
  - CT - Computed tomography → X-ray transmission
  - CBCT - Cone beam computed tomography → X-ray transmission
  - MRI - Magnetic resonance imaging → magnetic fields
  - US - Echography → ultrasound waves
  - SPECT - Single photon emission computed tomography → gamma ray emission
  - PET - Positron emission tomography → gamma ray emission
- Why multiple modalities?
  - Different purposes: anatomical vs. functional.
  - Different properties: resolution, contrast, etc.



Credits: B. Macq, J. Lee, F. Peeters, G. Kerckhofs (LGBIO2050 @ UCLouvain)

Electromagnetic spectrum



- Wavelength (in vacuum): $\lambda = {}^{c}/_{f}$, where:
  - $c = 299,792,458[m/s]$ is the speed of light in vacuum.
  - f is the frequency of the wave $[Hz] = [1/s]$.
  - Unit: meter.
- Photon energy: $E = h \cdot f$, where:
  - $h = 6.626 \cdot 10^{-34}[J \cdot s]$ is Planck's constant.
  - Unit: Joule $[J]$.
  - Electronvolt $[eV] = 1,6 \cdot 10^{-19}[J]$: Amount of kinetic energy gained by a single electron accelerating from rest through an electric potential difference of one volt in vacuum.
  - Photon energy expressed in electronvolts: $E \approx 4,1 \cdot 10^{-15} \cdot f$.
- *Don't learn those figures ;)*

https://en.wikipedia.org/wiki/Electromagnetic_spectrum

DICOM

DICOM
Digital Imaging and Communications in Medicine

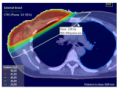
*"One **OPEN standard** to rule them all..."*


PET


CT


MRI


PACS (Picture Archiving and Communication System)

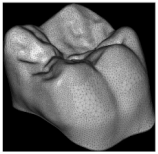


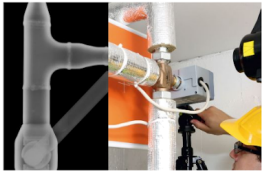

Radiotherapy

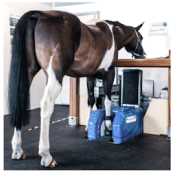
Everything is driven by software using the DICOM open standard!

Also “small imaging” and outside of hospitals!

UCLouvain





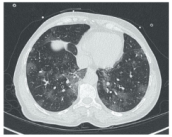


DICOM is everywhere!

DICOM
Digital Imaging and Communications in Medicine

File format

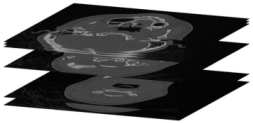
Acquisition data
(kind of XML)

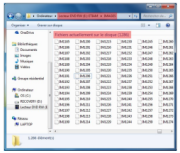


One slice of image
(kind of PNG)

Network protocol

Early example of Web service:
« store / query / retrieve »





Large variations in the volume of one medical image: radiography – 10MB, mammography – 100MB, CT-scan – 500MB, histology – 10GB

- "Digital Imaging and Communications in Medicine".
- De-facto open standard for digital medical imaging in all hospitals around the world.
- First normalized in 1985 by the ACR (*American College of Radiology*) and NEMA (*National Electrical Manufacturers Association*).
- Goal: Maximal technical interoperability between hardware and software, independently of their vendors.
- This session is about the DICOM file format.

Free and open-source software for DICOM

UCLouvain

Libraries

DCMTK

GDCM

VTK ITK

CORNERSTONE.JS

dcm4che.org

Viewers

Open Health Imaging Foundation

3DSlicer

WEASIS
medical viewer

Starviewer
Medical Imaging Software

Servers

dcm4chee

Dicoogle

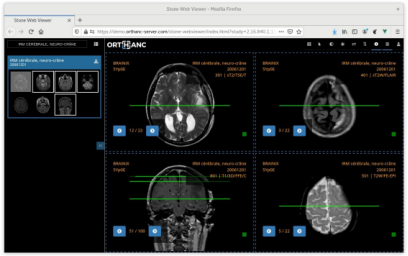
CONQUEST

ORTHANC

digital wallonia .be

Ancillary PACS for research or automation (yet main PACS)

Inter-site exchanges (including to the cloud)



Teleradiology portals

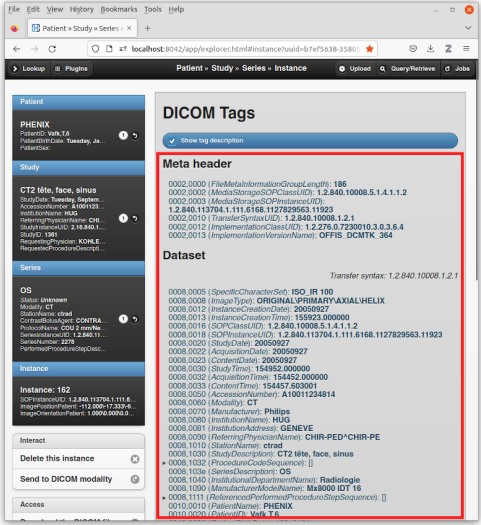
Structure of a DICOM file ("instance")

- A DICOM file is made of three parts:
  - The "meta-header" that identifies the file,
  - The "dataset" that contains the acquisition data,
  - The "pixel data" that contains the actual 2D image.
- The pixel data can be compressed according to a "*transfer syntax*":
  - In the clinical workflow, no compression (raw data) for maximum compatibility.
  - Lossless (non-destructive) compression:
    - RLE, JPEG2k, JPEG-LS.
    - Default choice for medical traceability.
  - Destructive compression:
    - JPEG.
    - Used for teleradiology (low-resolution image to reduce network bandwidth), and possibly digital pathology or visible-light imaging.
- A DICOM file can be multi-frame:
  - Several 2D images are concatenated inside the pixel data.
  - Commonly used by US devices (cine), PET-scanners and radiotherapy doses.

https://book.orthanc-server.com/dicom-guide.html#dicom-file-format

- The meta-header and the dataset are lists of DICOM tags associated with values:
  - A DICOM tag is a pair of hexadecimal numbers (up to 65535): The group and the element.
  - The value associated with a DICOM tag has a data type: the value representation.
  - The DICOM standard lists mandatory ("types 1 and 2") or optional ("type 3") DICOM tags for each imaging modality.
  - DICOM tags can be nested (recursive structure like HTML or JSON) → "sequences".
- The DICOM standard defines a nomenclature of DICOM tags, for instance:
  - `(0x0010,0x0010)` → Patient's name
  - `(0x0010,0x0030)` → Patient's birth date
  - `(0x0008,0x1030)` → Study description
  - `(0x0008,0x1010)` → Station name
  - `(0x0008,0x0060)` → Modality (MR, CT, RX, US...)
  - `(0x0028,0x0010)` and `(0x0028,0x0011)` → Rows and columns (height/width)
- The pixel data is actually part of the dataset, as the `(0x7fe0,0x0010)` tag.

https://book.orthanc-server.com/dicom-guide.html#dicom-file-format



NB: The command-line tools “`dcm2xml`” (from DCMTK) and “`gdcmxml`” (from GDCM) can also be used.

Extremely simple: Concatenate row by row, then frame by frame!

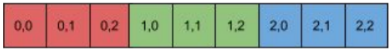


Image of size “3x3” (single-frame)

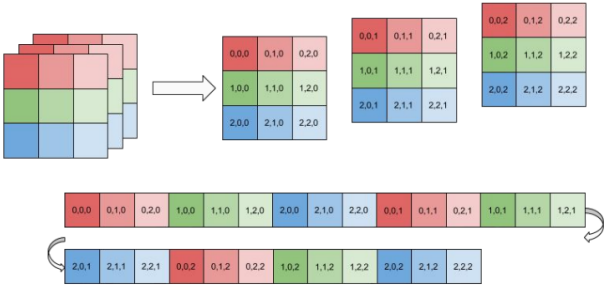
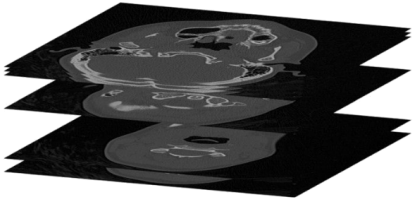
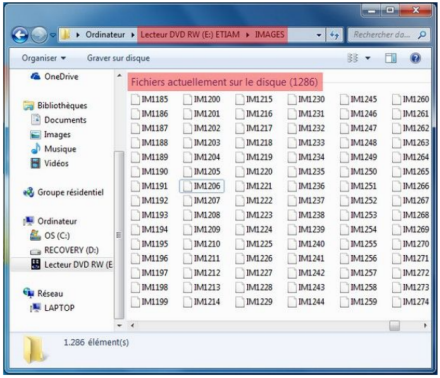


Image of size “3x3x3” (multi-frame)

An imaging study/series = a set of DICOM instances (files)

UCLouvain





Large variations in the volume of one imaging study:

- radiography ~ 10MB (2 files of 5MB),
- mammography ~ 100MB (4 files of 30MB),
- CT-scan ~ 500MB (1000 files of 500KB),
- MRI ~ 1GB (large variations).

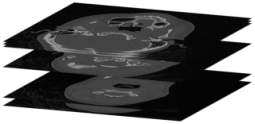
- 3D volumes are typically encoded as multiple DICOM instances (one per 2D slice).

From instances to patients

UCLouvain

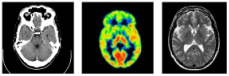
One DICOM instance

Series



Group of instances generated by the **same modality** during the same acquisition

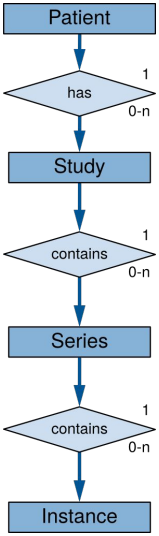
Study



Group of series generated during the **same clinical examination** of the patient

Patient

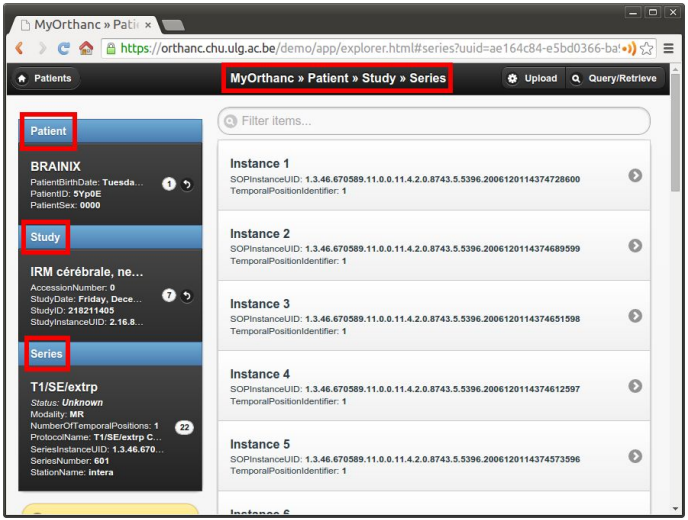
Group of studies generated throughout patient's life (**longitudinal**)

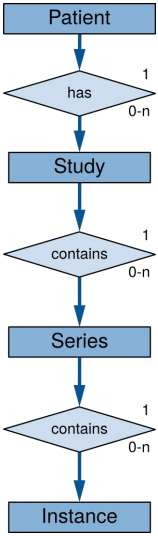


- "Russian dolls" model.
- "Studies" vs. "series":
  - Nuclear medicine → two series (CT + PET) that are part of the same study.
  - CT-scanner → multiple series (scout, sagittal, coronal, axial, preview, dose reports...).
  - MRI → multiple series depending on the acquisition sequence (T1, T2, FLAIR, BOLD...).
- DICOM specifies one module (set of DICOM tags) for each of the "Patient", "Study", "Series" and "Instance" levels.

DICOM hierarchy, as displayed by Orthanc

UCLouvain





- Each DICOM instance contains 4 identifier tags:
  - `(0x0010,0x0020)` → Patient ID (in the "Patient" module)
  - `(0x0020,0x000d)` → Study Instance UID (in the "Study" module)
  - `(0x0020,0x000e)` → Series Instance UID (in the "Series" module)
  - `(0x0008,0x1018)` → SOP Instance UID (in the "Instance" module)
- Those can be used as primary keys for the database of a PACS.
- Uniqueness:
  - Study/Series/SOP instance UID are guaranteed to be globally unique (i.e. all around the world).
  - Patient ID is only unique locally unique (inside the hospital) → need for reconciliation if medical images are transferred from one hospital to another.

Table A.3-1. CT Image IOD Modules

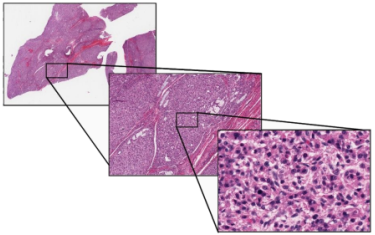
IE	Module	Reference	Usage
Patient	Patient	C.7.1.1	M
	Clinical Trial Subject	C.7.1.3	U
Study	General Study	C.7.2.1	M
	Patient Study	C.7.2.2	U
	Clinical Trial Study	C.7.2.3	U
Series	General Series	C.7.3.1	M
	Clinical Trial Series	C.7.3.2	U
Frame of Reference	Frame of Reference	C.7.4.1	M
Equipment	General Equipment	C.7.5.1	M
Image	General Image	C.7.6.1	M
	Image Plane	C.7.6.2	M
	Image Pixel	C.7.6.3	M
	Contrast/Bolus	C.7.6.4	C - Required if contrast media was used in this image
	Device	C.7.6.12	U
	Specimen	C.7.6.22	U
	CT Image	C.8.2.1	M
	Overlay Plane	C.9.2	U
	VOI LUT	C.11.2	U
	SOP Common	C.12.1	M
	Common Instance Reference	C.12.2	U

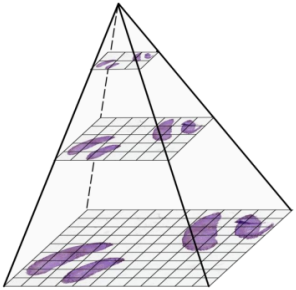
Table C.7-3. General Study Module Attributes

Attribute Name	Tag	Type	Attribute Description
Study Instance UID	(0020,000D)	1	Unique identifier for the Study.
Study Date	(0008,0020)	2	Date the Study started.
Study Time	(0008,0030)	2	Time the Study started.
Referring Physician's Name	(0008,0090)	2	Name of the patient's referring physician
Study ID	(0020,0010)	2	User or equipment generated Study identifier.
Accession Number	(0008,0050)	2	A RIS generated number that identifies the order for the Study.

- DICOM standard groups DICOM tags by **modules**.
- Each type of imaging modality is associated with a **IOD** that lists the mandatory and optional modules.
- One module can be shared by multiple IOD.

- A module can be mandatory (M), optional (U) or conditional (C) in an IOD.





DICOM encoding
(cf. Google Maps)

- Very large 2D images → 100,000 x 100,000 pixels → up to 20GB.
- Different zoom levels are precomputed (**multi-scale pyramid**).
- Stored as many multi-frame files, each tile being one frame.

  - A single tile will typically be 256×256 or 512×512 pixels.

https://wsi.orthanc-server.com/demo/

```java
import java.io.FileInputStream;
import java.io.IOException;
import org.dcm4che3.data.Attributes;
import org.dcm4che3.data.Tag;
import org.dcm4che3.io.DicomInputStream;

public class Test {
    public static void main(String[] args) throws IOException {
        FileInputStream stream = new FileInputStream("hand.dcm");

        Attributes dataset;

        try (DicomInputStream din = new DicomInputStream(stream)) {
            dataset = din.readDataset();
        }

        System.out.println("Patient name: " + dataset.getString(Tag.PatientName));
        System.out.println("Study date: " + dataset.getString(Tag.StudyDate));
    }
}
```

- Java applications will typically use the dcm4che library.
- Class `be.uclouvain.DicomImage` provides an easy-to-use wrapper.

HTML5 canvas

From HTML:

```
<canvas id="my-canvas" width="640" height="480"></canvas>
```

Our BEST Rendering Library has to wrap `<canvas>` within a `<div>` to control its size using CSS:

```
<div id="my-drawing" class="container">
  <canvas></canvas>
</div>
```

From JavaScript:

```
var canvas = document.createElement('canvas');
canvas.width = 640;
canvas.height = 480;
```

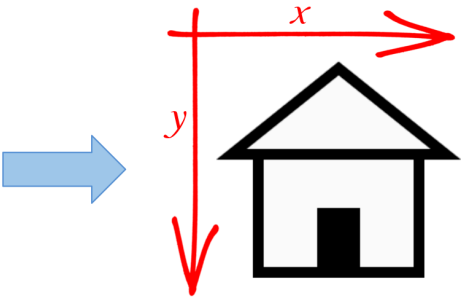
- This creates an invisible “offscreen” canvas
- `canvas` is typically a global variable

Drawing on a canvas is done by getting its **rendering context**:

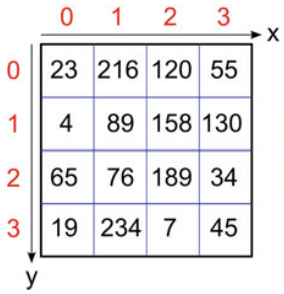
```
// HTML canvas
var ctx = document.getElementById('my-canvas').getContext('2d');

// Offscreen canvas
var canvas = document.createElement('canvas');
canvas.width = 640;
canvas.height = 480;
var ctx = canvas.getContext('2d');
```

https://developer.mozilla.org/en-US/docs/Web/API/CanvasRenderingContext2D

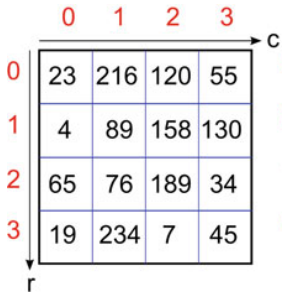
```
// Set line width
ctx.lineWidth = 10;

// Wall
ctx.strokeRect(75, 140, 150, 110);

// Door
ctx.fillRect(130, 190, 40, 60);

// Roof
ctx.beginPath();
ctx.moveTo(50, 140);
ctx.lineTo(150, 60);
ctx.lineTo(250, 140);
ctx.closePath();
ctx.stroke();
```



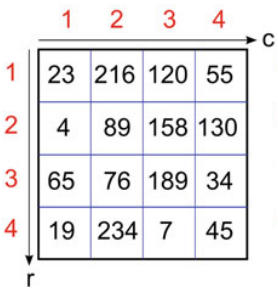
https://developer.mozilla.org/en-US/docs/Web/API/CanvasRenderingContext2D



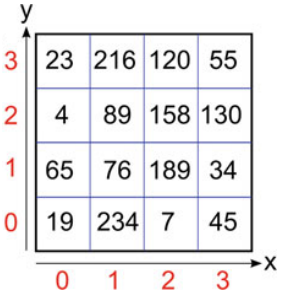
DICOM / HTML5 / `BufferedImage` in Java



numpy in Python / `RealMatrix` in Java



Matlab / Octave



Traditional mathematical coordinates

https://doi.org/10.1007/978-3-030-39364-9 (Section 2.4.1)

Controlling style

UCLouvain

```
// Change color of stroke (lines)
ctx.strokeStyle = 'red';

// Change color of filling
ctx.fillStyle = 'gray';

// Drawing a circle
ctx.beginPath();
ctx.arc(centerX, centerY, 100, 0, 2.0 * Math.PI);
ctx.stroke();

// Filling a circle
ctx.beginPath();
ctx.arc(centerX, centerY, 100, 0, 2.0 * Math.PI);
ctx.fill();
```

https://developer.mozilla.org/en-US/docs/Web/API/CanvasRenderingContext2D

Copying an image

UCLouvain

From an offscreen canvas:

```
<canvas id="my-canvas" width="640" height="480"></canvas>
```

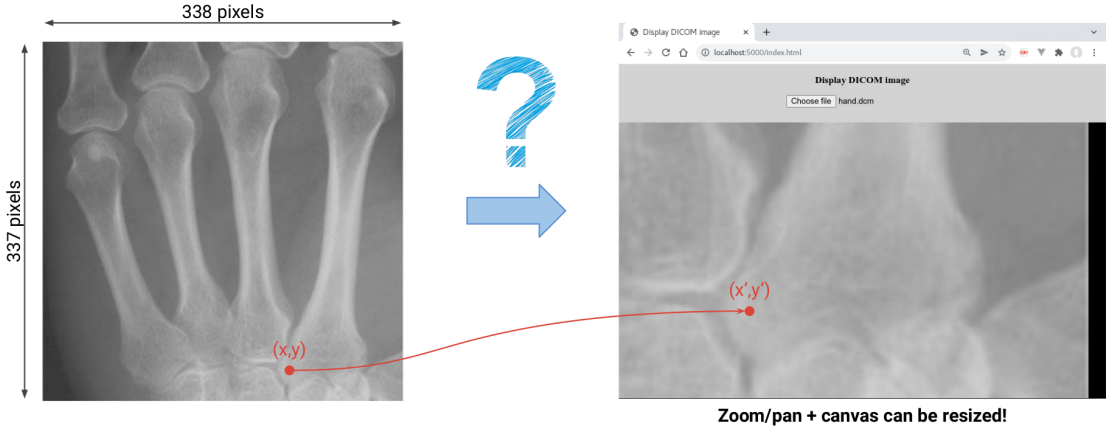
```
var canvas = document.createElement('canvas');

// ...Draw onto the offscreen canvas...

var ctx = document.getElementById('my-canvas').getContext('2d');
ctx.drawImage(canvas, x, y);
```

From a PNG/JPEG “sprite” loaded by HTML:

```
<canvas id="my-canvas" width="640" height="480"></canvas>
<img id="sprite" src="sprite.png" style="display:none" />
```

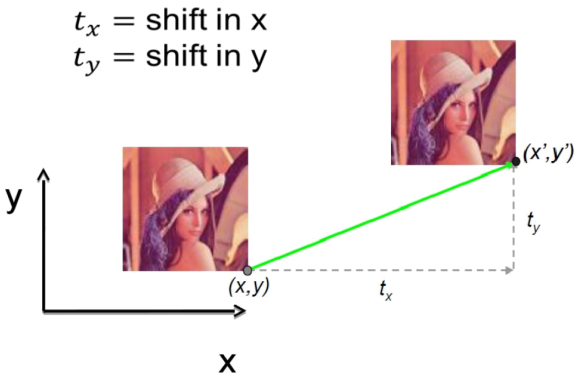
```
var sprite = document.getElementById('sprite');
var ctx = document.getElementById('my-canvas').getContext('2d');
ctx.drawImage(sprite, x, y);
```

From image to canvas coordinates



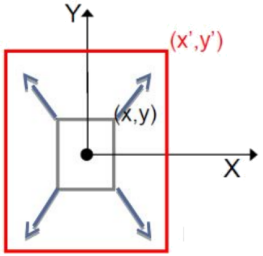
Zoom/pan + canvas can be resized!

Pan/translation = addition



$$\begin{cases} x' = x + t_x \\ y' = y + t_y \end{cases}$$

s_x = scaling factor in x
s_y = scaling factor in y



$$\begin{cases} x' = s_x \cdot x \\ y' = s_y \cdot y \end{cases}$$

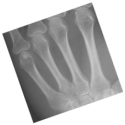
- NB: Before applying zoom, the center of the drawing should generally be translated at coordinate $(0, 0)$.

zoom

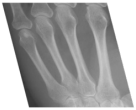
translation

$$\begin{cases} x' = a_{11} \cdot x + a_{12} \cdot y + a_{13} \\ y' = a_{21} \cdot x + a_{22} \cdot y + a_{23} \end{cases} \Leftrightarrow \begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \underbrace{\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ 0 & 0 & 1 \end{bmatrix}}_{A \in \mathbb{R}^{3\times 3}} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

“2D homogeneous coordinates”



rotation



shearing

Chaining transformations

Matrix multiplication

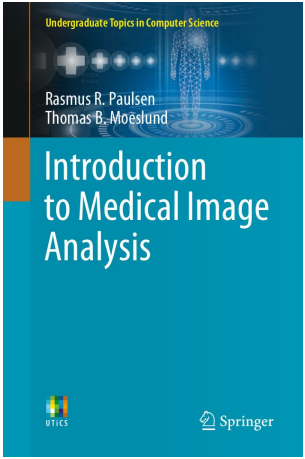
$$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = A \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}; \begin{pmatrix} x'' \\ y'' \\ 1 \end{pmatrix} = A' \begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix}; \begin{pmatrix} x''' \\ y''' \\ 1 \end{pmatrix} = A'' \begin{pmatrix} x'' \\ y'' \\ 1 \end{pmatrix}$$

$$\Rightarrow \begin{pmatrix} x''' \\ y''' \\ 1 \end{pmatrix} = \boxed{A''A'A} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

Reversing a transformation

Matrix inversion

$$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = A \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} \Rightarrow \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \boxed{A^{-1}} \begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix}$$


Undergraduate Topics in Computer Science
Rasmus R. Paulsen
Thomas B. Moeslund
Introduction to Medical Image Analysis
UTiCS
Springer

Chapter 10

Auxiliary function:

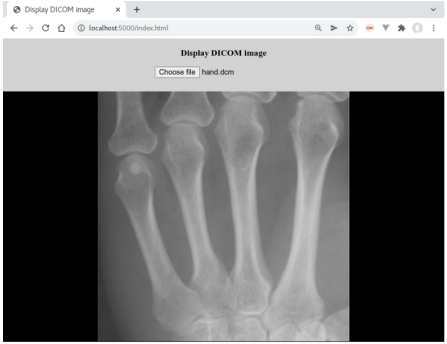
```
var CreateTransform = function(a11, a12, a13,
                               a21, a22, a23) {
  // "DOMMatrix" uses a transposed matrix
  return new DOMMatrix([ a11, a21,
                         a12, a22,
                         a13, a23 ]);
};
```

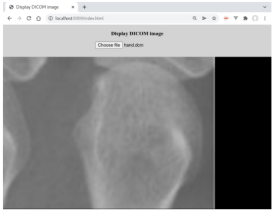
```
var ctx = document.getElementById('my-canvas').getContext('2d');
var backup = ctx.getTransform();

var zoom = CreateTransform(2, 0, 0,
                           0, 2, 0);
ctx.setTransform(zoom);

var pan = CreateTransform(1, 0, 100,
                          0, 1, 200);
ctx.setTransform(pan);

// This is the matrix product "zoom * pan"
var firstPanThenZoom = zoom.multiply(pan);
ctx.setTransform(firstPanThenZoom);

ctx.setTransform(backup);  // Back to the original transformation
```

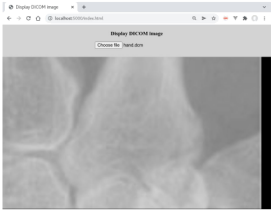
JavaScript library developed as a support to the BEST summer course "Hacking medical imaging" in Louvain-la-Neuve (July 2022).

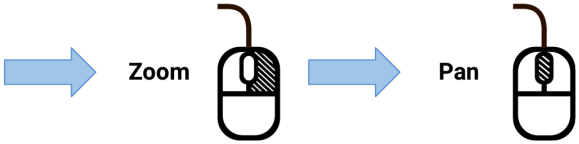
Mouse conventions in radiology

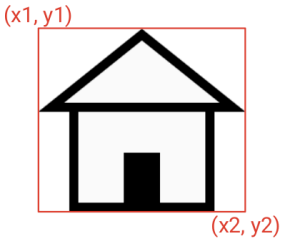
UCLouvain





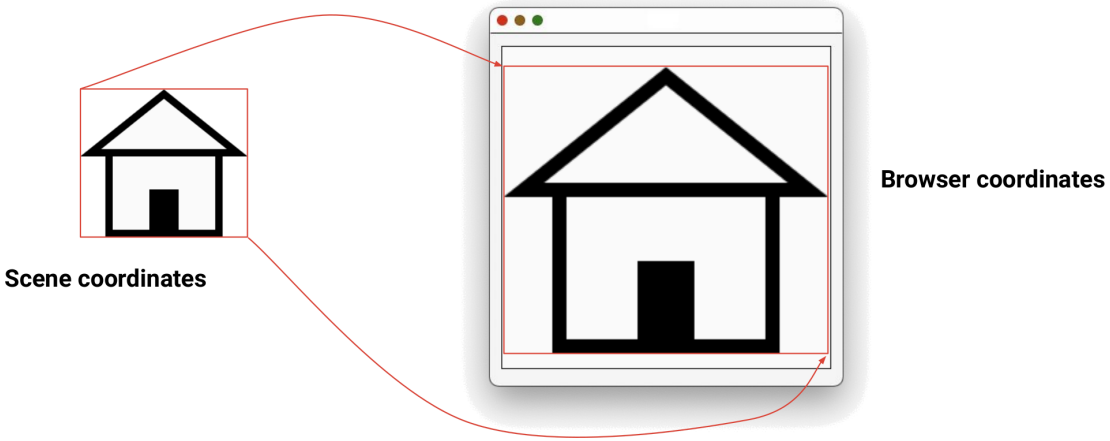


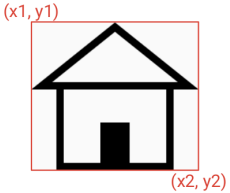




Scene coordinates

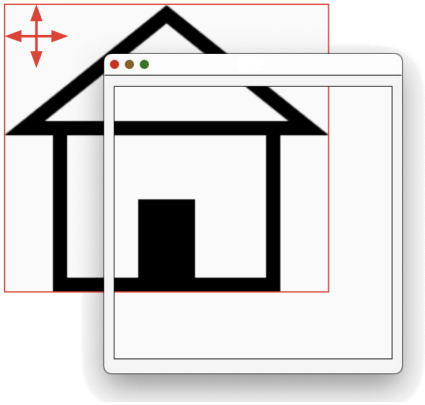
The designer of a Web application for medical imaging wants to focus on drawing the **content of scene**, not on the coordinates of the HTML page!


Scene coordinates
Browser coordinates



Scene coordinates

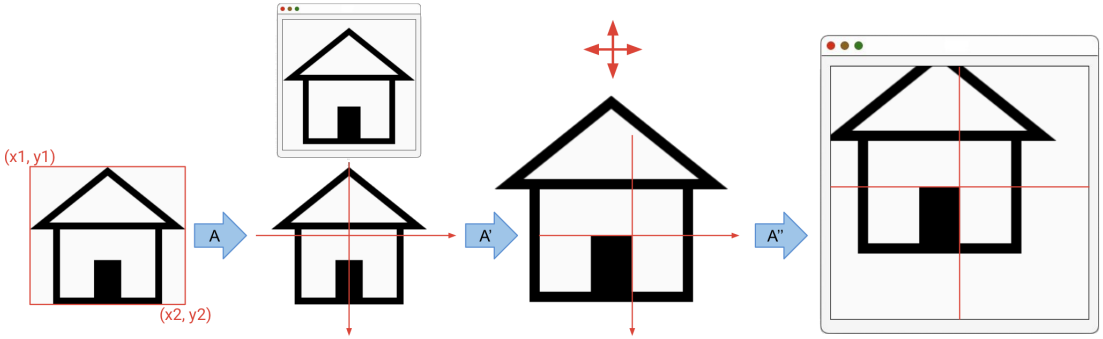
Viewport coordinates (zoom/pan)



Browser coordinates
(size of the window)

Chain of 3 affine transformations

UCLouvain



Scene coordinates
(developer)

Initial viewport
(fit extents)

User viewport
(zoom/pan)

Browser coordinates
(origin-to-center)

The BEST rendering library manages this for you!

NB: Don't hesitate to look at the source code "`BestRendering.js`" to know how transformations and mouse events are handled!


Scene coordinates
(-1, -1)
(0.5, 0)
(2, 1)
“f”
“g”
“r”

Example (2/4): Drawing a simple scene

UCLouvain

index.html

```html
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <style>
      html, body {
          width: 100%;
          height: 100%;
          margin: 0px;
          border: 0;
          overflow: hidden; /*  Disable scrollbars */
          display: block;  /* No floating content on sides */
      }
      #my-drawing {
          position: absolute;
          left: 0;
          top: 0;
          width: 100%;
          height: 100%;
      }
    </style>
  </head>
  <body>
    <div id="my-drawing" class="container">
      <canvas></canvas>
    </div>
    <script src="BestRendering.js"></script>
    <script src="app.js"></script>
  </body>
</html>
```

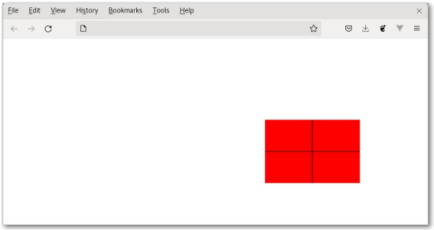
app.js

```js
var color = 'red';

function GetExtent() {
  // Define the extent of the scene, i.e. (x1,y1) and (x2,y2)
  return BestRendering.CreateExtent(-1, -1, 2, 1);
}

function Draw(ctx) {
  // "ctx" is now working in scene coordinates!
  ctx.fillStyle = color;
  ctx.fillRect(-1, -1, 3 /* width */, 2 /* height */);

  ctx.strokeStyle = 'black';
  ctx.moveTo(0.5, -1); ctx.lineTo(0.5, 1);  ctx.stroke();
  ctx.moveTo(-1, 0);   ctx.lineTo(2, 0);    ctx.stroke();
}

BestRendering.InitializeContainer('my-drawing', Draw, GetExtent);
```

Example (3/4): Re-fit the canvas after zoom/pan

UCLouvain

```
document.addEventListener('keydown', function(event) {
  if (event.key == 'f') {
    BestRendering.FitContainer('my-drawing');
  }
});
```



Example (4/4): Redraw after a change in the scene

UCLouvain

```
document.addEventListener('keydown', function(event) {
  if (event.key == 'g') {
    color = 'green';
    BestRendering.DrawContainer('my-drawing');
  } else if (event.key == 'r') {
    color = 'red';
    BestRendering.DrawContainer('my-drawing');
  }
});
```


UCLouvain