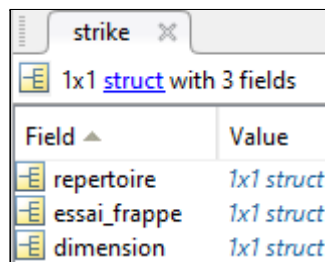


# Description of the MATLAB program

## *Matlab\_Script\_for\_Cumputed\_data.m*

The Zenodo repository contains zip files for data (Data.zip) and Matlab functions library (Library.zip), and the Matlab script (Matlab\_Script\_for\_Cumputed\_data.m) used in the Famié et al PLOSONE study. In addition, the present *README FIRST MATLAB program.pdf* file explains the Matlab code and data organization.

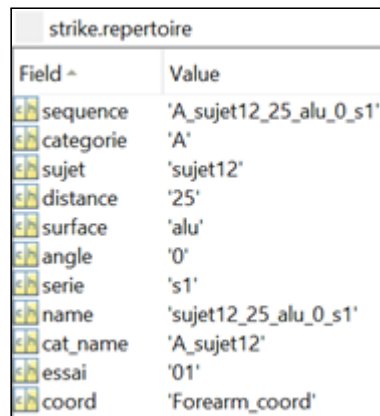
### A) Data preprocessing



Field	Value
repertoire	1x1 struct
essai_frappe	1x1 struct
dimension	1x1 struct

Fig 1

The *Data/1\_Raw\_Data* directory contains the initial segmented data files of each participant (33 subjects x 120 trials = 3960 MATLAB files). The MATLAB file for a single trial is a structure called “strike” that embeds 3 other sub-structures: repertoire, essai\_frappe and dimension (see Fig1). Figure 2 illustrates *A\_sujet12\_25\_alu\_0\_s1\_E01.mat* file with the “repertoire” sub-structure that contains all the necessary data information (Fig 1).



Field	Value
sequence	'A_sujet12_25_alu_0_s1'
categorie	'A'
sujet	'sujet12'
distance	'25'
surface	'alu'
angle	'0'
serie	's1'
name	'sujet12_25_alu_0_s1'
cat_name	'A_sujet12'
essai	'01'
coord	'Forearm_coord'

Fig 2

The “essai\_frappe” sub-structure (see Fig 3 below) contains the “data\_f” matrix with the XYZ positions for all the markers. Then, the data for each marker has been isolated in a separate sub-structure which contains four data vectors: x, y, z, p\_cart (Cartesian position =  $p\_cart = \sqrt{x^2+y^2+z^2}$ ) and the xyz matrix (see Fig 4 for the I\_P\_D marker). Although the data contains a huge number of markers (Fig 3), half of dozens were used for the subsequent data analysis (see section B - Data processing).

strike.essai_frappe					
Field ^	Value				
data_f	244x70 double		Rad_styl	1x1 struct	: Radius styloid
I_P_D	1x1 struct	: Index phalanx distal	Ulna_styl	1x1 struct	: Ulba styloid
I_P_M	1x1 struct	: Index phalanx medial	E_Lateral	1x1 struct	: Epicondyle Lateral
I_P_P	1x1 struct	: Index phalanx proximal	E_Medial	1x1 struct	: Epicondyle Medial
I_M_T	1x1 struct	: Index metacarp Head	Acro_1	1x1 struct	: Acromion right
I_M_B	1x1 struct	: Index phalanx base	Acro_2	1x1 struct	: Acromion left
P_P_D	1x1 struct	: Thumb phalanx distal	Sternum	1x1 struct	: Sternum
P_P_P	1x1 struct	: Thumb phalanx proximal	Cube_A	1x1 struct	: Marker on the middle of the cube front edge
P_M_T	1x1 struct	: Thumb metacarp Head	Cube_D	1x1 struct	: Marker on the right back edge of cube
Maj_M_T	1x1 struct	: Midle finger metacarp Head	Cube_G	1x1 struct	: Marker on the left back edge of cube
Maj_M_B	1x1 struct	: Midle finger metacarp Base	Debut_D	1x1 struct	: Marker on the right start cube initial position on the setup
			Debut_G	1x1 struct	: Marker on the left start cube initial position on the setup
			Cible	1x1 struct	: Target distance

Fig 3 Markers used during the motion capture which are contained in the *data\_f* matrix

strike.essai_frappe.I_P_D		
Field ^	Value	
x	238x1 double	
y	238x1 double	
z	238x1 double	
xyz	238x3 double	
p_cart	238x1 double	

Fig 4

The “*dimension*” sub-structure (see Fig 1 and 5) contains a “Sujet” sub-structure that includes two other sub-structures (namely, info\_P and Info\_Sujet) containing information about participant’s characteristics. “Sujet.info.P” contains participant’s number, sex (‘M’ male or ‘F’ female), age, height, and mass (in Kg), as illustrated in Fig 5.

strike.dimension.Sujet		
Field ^	Value	
info_P	1x1 struct	
Info_Sujet	1x11 struct	

strike.dimension.Sujet.info_P		
Field ^	Value	
number	12	
sexe	'M'	
age	24	
hight	188	
mass	85	

Fig 5

“Sujet.Info\_Sujet” (see Fig 6) contains anthropometric data about the participant's forearm and hand that were motion captured just before the experiment started. The length of the segments (*Seg\_norme*) was computed along the lines of ISBN (Wu et al, 2005). The Center of Mass (*CM*) and the radius of gyration (*Rg*) for the x, y and z axes were computed for forearm and hand segments using the DeLeva (1996) anthropometric table. The mass of each segment was estimated from the total mass of the participant (see Fig 5).

strike.dimension.Sujet.Info_Sujet				
Fields	Variables	Forearm	Hand	Unite
1	'Seg_norme'	0.2802	0.0943	'm'
2	'CM_DeLeva'	0.4574	0.7900	'%
3	'CM_norme'	0.1282	0.0745	'm'
4	'Masse_DeLeva'	0.0162	0.0061	'%
5	'Masse_seg'	1.3770	0.5185	'kg'
6	'Rg_x_DeLeva'	0.2650	0.5130	'%
7	'Rg_x'	0.0743	0.0484	'Unite'
8	'Rg_y_DeLeva'	0.1210	0.4010	'%
9	'Rg_y'	0.0339	0.0378	'Unite'
10	'Rg_z_DeLeva'	0.2760	0.6280	'%
11	'Rg_z'	0.0773	0.0592	'Unite'

Fig 6 Forearm and hand parameters for subsequent biomechanical computations

## B) Data processing

Note that the output files (for each trial of each individual) of the data processing described hereafter is stored in the *Data/2\_Computed\_Data* directory. These files are intermediate stages of processing leading to the *Data/3\_Data\_for\_stats* directory containing a synthetic file (in two different formats) with the data used for the stats.

```

%% Download file
[ filename ] = nom_fichier_essai( loop_param.categories(i), loop_param.subjects(j,:), loop_param.series(k), loop_param.E(m,:));
load([CurrentFolder, '\Data\1_Raw_Data\', filename]);

%% Data calculations
[ strike.essai_frappe ] = new_markers_v1( strike.essai_frappe );
[ strike ] = filt_derive_X_Xdot_Xddot_v1( strike );
[ strike.vect ] = vect_exp_v1( strike.essai_frappe );
[ strike.angle ] = angles_2D_exp_v1( strike.vect );
[ strike.parametre.time, strike.parametre.conversion ] = time_conversion_v1( strike.essai_frappe.I_P_D.y );
[ strike.timing ] = v_cart_all_v2( strike ); % v_cart_all_v1
[ strike ] = phase_mouvement_v2( strike, strike.repertoire.essai );
[ strike ] = KE_Winter(strike);

```

Fig 7. Excerpt from the “Matlab\_Script\_for\_Computed\_data.m” script

The Data processing was run with the “Matlab\_Script\_for\_Computed\_data.m” matlab script (see Fig 7) that loops 1) data import (see %% Download file section), and 2) computations (see %% Data calculations section).

%% Data calculations section contains all the processing functions applied to the “strike” structure (see Fig 1) containing all the data for a given trial.

The new\_marker\_v1 function computes the elbow articular center using the center\_markers\_v1 function from the *Library* directory. center\_markers\_v1 computes the mid-distance between lateral (E\_Lateral) and medial (E\_Medial) elbow markers positions, and stored it in the “essai\_frappe” sub-structure (see Fig 1 above).

The `filt_derive_X_Xdot_Xddot_v1` function computes the derivatives based on the `data_f` matrix to obtain the velocity and the acceleration of each marker. The output is added to the “strike” structure (see Fig 1 and Fig 8), as it will also be the case for subsequent outputs.

1x1 struct with 10 fields	
Field ▲	Value
repertoire	1x1 struct
essai_frappe	1x1 struct
dimension	1x1 struct
deriv	1x1 struct
vect	1x1 struct
angle	1x1 struct
parametre	1x1 struct
timing	1x1 struct
mvt_param	1x1 struct
EC	1x1 struct

Fig 8

The `vect_exp_v1` function creates a structure containing several vectors (see Fig 9), namely two vectors per segment (one vector oriented toward the proximal joint, and the other oriented toward the distal joint).

`vect_seg` uses the xyz matrix of two markers and does the subtraction between the two matrices to compute the two vectors in opposite directions (see previous sentence). Within the `vect_seg` function, `vect_param` isolates each x, y, z coordinate in a vector, and computes the projection onto the xy horizontal plane. Within the `vect_seg` function, `vect.norme` computes the length of the vector in mm using the `cart` function ( $cart = @(x,y,z) \sqrt{x.^2 + y.^2 + z.^2};$ ). The vector functions are available from the *Vector* sub-directory of the *Library* directory.

```

1 function [ vect ] = vect_exp_v1( essai )
2 %UNTITLED4 Summary of this function goes here
3 % Detailed explanation goes here
4
5 %% Vecteurs
6
7 %Index finger
8 % Index finger from the distal phalanx to the index metacarp head
9 [ vect.IPD_IMT, vect.IMT_IPD ] = vect_seg( essai.I_P_D.xyz, essai.I_M_T.xyz);
10 %Metacarp
11 % Index phalang distal marker with the index metacarp head
12 [ vect.IMT_IMB, vect.IMB_IMT ] = vect_seg( essai.I_M_T.xyz, essai.I_M_B.xyz);
13 % The middle finger metacarp head with the the middle finger metacarp base
14 [ vect.MajMT_MajMB, vect.MajMb_MajMT ] = vect_seg( essai.Maj_M_T.xyz, essai.Maj_M_B.xyz);
15 %Metacarp to Radius styloid
16 % Index metacarp head with Radius styloid
17 [ vect.IMT_Radstyl, vect.Radstyl_IMT ] = vect_seg( essai.I_M_T.xyz, essai.Rad_styl.xyz);
18 %Forearm
19 % Radius styloid with elbow articular center
20 [ vect.Radstyl_CAcoude, vect.CAcoude_Radstyl ] = vect_seg( essai.Rad_styl.xyz, essai.CA_coude.xyz);
21 % Beginning ligne between the right and left marker with symbolise the start position on the setup
22 [ vect.DebutD_DebutG, vect.DebutG_DebutD ] = vect_seg( essai.Debut_D.xyz, essai.Debut_G.xyz);
23
24 end

```

Fig 9 Computed vectors

angles\_2D\_exp\_v1 function (see Fig 10 below) computes several angles projected onto the horizontal plane. The angle\_seq sub-function computes for each time frame the angle between two vectors, namely the index finger angle (angle.IPD\_IMT\_Radstyl), the wrist angle (angle.IMT\_Radstyl\_CAcoude) and the forearm angle (angle.elbow). angle\_seq also computes angular speed and acceleration in radian and degree.

```

1 function [ angle ] = angles_2D_exp_v1( vect )
2 %UNTITLED5 Summary of this function goes here
3 % Detailed explanation goes here
4 %% Angle
5 %2D
6 [ angle.IPD_IMT_Radstyl.D2 ] = angle_seq( vect.IMT_Radstyl.xy_z0, vect.IMT_IPD.xy_z0 );
7 [ angle.IMT_Radstyl_CAcoude.D2 ] = angle_seq( vect.Radstyl_CAcoude.xy_z0, vect.Radstyl_IMT.xy_z0 );
8 [ angle.elbow.D2 ] = angle_seq( vect.CAcoude_Radstyl.xy_z0, vect.DebutG_DebutD.xy_z0 );
9 end

```

Fig 10

time\_conversion\_v1 stores additional information in the *parametre* sub-structure of the “strike” structure (see Fig 8), namely the number of frames (*nb\_frames*), the duration of movement (*end\_times* in seconds), etc. (see Fig 11).

strike.parametre.time.x	
Field ▲	Value
nb_frames	272
end_times	1.0880
frames	272x1 double
times	272x1 double

Fig 11

v\_cart\_all\_v2 computes several parameters relative to the different phases of movement (arming, striking, cube sliding) based on the speed of the index fingertip and of the cube (see manuscript Figure 2).

```

function [ modification ] = v_cart_all_v2( strike )
%UNTITLED Summary of this function goes here
% Detailed explanation goes here
%%
%% Cube Speed
%%
%% Y
[modif] = Y_speed_param(strike);
%%
%% Cart
%%
[modif] = Cart_speed_param(strike,modif);
%%
%% Data Selected
%%
[modification] = Speed_parameter_selected(strike,modif);
end

```

Fig 12

The `Y_speed_param` sub-function (see Fig 12) uses speed along the Y axis (direction of travel of the cube in the gutter towards the target) to compute the following parameters :

- cube maximum speed (corresponding to the end of contact between finger and cube);
- the frame number just before the cube starts to move;
- maximal fingertip speed;
- the frame number when the strike is initiated;
- and fingertip speed before the cube starts to move.

`Cart_speed_param` (see Fig 12) converts some outputs of `Y_speed_param` into cartesian speed.

`Speed_parameter_selected` (see Fig 12) selects the optimal parameters for data analysis (see Fig 13), using the MSI method explained in the manuscript :

- cube maximum cartesian speed;
- the frame number before just before the cube starts to move;
- maximal fingertip cartesian speed;
- the frame number when the strike is initiated;
- the frame just before the contact between the index finger tip and the cube (using the MSI technique described in the manuscript, and implemented in `MSI_tech_VIPD_v1`).

strike.timing.final	
Field ▲	Value
Cube_A_cart_max_ind	188
Cube_A_cart_max_val_mmF	3.9976
Cube_A_cart_max_val_ms	0.9994
Cube_A_cart_start_ind	177
Cube_A_cart_start_val_mmF	0.0941
Cube_A_cart_start_val_ms	0.0235
IPD_cart_max_ind	176
IPD_cart_max_val_mmF	6.3877
IPD_cart_max_val_ms	1.5969
IPD_cart_start_ind	110
IPD_cart_start_val_mmF	0
IPD_cart_start_val_ms	0
IPD_cart_contact_ind	177
IPD_cart_contact_val_mmF	6.2507
IPD_cart_contact_val_ms	1.5627

Fig 13

---

The `phase_mouvement_v2` function creates the `mvt_param` sub-structure (see Figure 14).

strike.mvt_param	
Field ▲	Value
categorie_num	1
categorie_name	'A'
coord_num	1
coord_name	'F'
sujet_num	12
sujet_name	'sujet12'
target_num	1
distance_name	'25'
material_num	1
surface_name	'alu'
slope_num	2
angle_name	'0'
serie_num	1
serie_name	's1'
trial_num	1
essai_name	'01'
Cube_V_max	0.9994
Finger_V_impact	1.5627
spatial_error	-32.5060
Finger_Angular_diff	9.0168
Wrist_Angular_diff	18.2993
Elbow_Angular_diff	6.0607

Fig 14

*strike.mvt\_param* contains information about participant and trial characteristics, for each trial (see Fig 14), namely: block order (*categorie*), subject number, target distance, surface material, surface slope, trial number, etc. Note that we added here a motor coordination (*coord*) parameter (F = Forearm; W = Wrist) *a posteriori* on the basis of the cluster analysis (see manuscript) in order to have this information also in the data files generated for the subsequent statistical analyses.

Finally *strike.mvt\_param* lists kinematics dependent variables such as: maximal cube speed (*Cube\_Vmax*) in m/s, fingertip speed at impact (*Finger\_V\_impact*), spatial error (*spatial\_error*) in percentage of target distance, and the angular amplitude in the horizontal plane for finger, wrist, and elbow (see Fig 14).

---

Finally, the *KE\_Winter* function (see Figure 15) computes the kinetic energy variables that will be added to the *mvt\_param* sub-structure (see Figure 16): *KE\_Forearm*, *KE\_Hand*, *KE\_Total*, *KE\_Cube*.

```

function [strike] = KE_Winter(strike)
%UNTITLED Summary of this function goes here
% Detailed explanation goes here

%% Calculé of Kinectic energy
% Stocker les dimensions DeLeva du sujet
% CM : position et Vecteur
[ strike ] = Code_CM_essai_v2( strike);
% CM : Recalcule vitesse angulaire radian / second
[ strike ] = angle_EC_mvt( strike );
% Vitesse dans le plan xy
[ strike.EC.CM.Xdot.Theo.V_Forearm ] = V_Forearm_Theo( strike );
[ strike.EC.CM.Xdot.Theo.V_Hand ] = V_Hand_Theo_3( strike );
% KE Winter
[ strike ] = EC_Winter_v1( strike );
% KE cube
[strike] = EC_Cube(strike);

end

```

Fig 15

In order to compute the KE variables (using Winter's model, 2009) for the upper limb, the `KE_Winter` function starts with computing the forearm and the hand mass center position during the strike using the `Code_CM_essai_v2` function. Then, the `angle_EC_mvt` function computes the joint angles necessary for the KE variables. The `V_Forearm_Theo` and `V_Hand_Theo_3` functions compute the center of mass speed of the forearm and hand. On the basis of the four previously mentioned functions and of the "dimension" sub-structure of the "strike" structure (see Figure 8), the `EC_Winter_v1` function computes the *KE\_Forearm*, *KE\_Hand*, *KE\_Total* variables (see Figure 16). Finally, the `EC_Cube` function computes cube KE at maximum cube speed.

strike.mvt_param	
Field ▲	Value
categorie_num	1
categorie_name	'A'
coord_num	1
coord_name	'F'
sujet_num	12
sujet_name	'sujet12'
target_num	1
distance_name	'25'
material_num	1
surface_name	'alu'
slope_num	2
angle_name	'0'
serie_num	1
serie_name	's1'
trial_num	1
essai_name	'01'
Cube_V_max	0.9994
Finger_V_impact	1.5627
spatial_error	-32.5060
Finger_Angular_diff	9.0168
Wrist_Angular_diff	18.2993
Elbow_Angular_diff	6.0607
KE_Forearm	0.0088
KE_Hand	0.1287
KE_Total	0.1375
KE_Cube	0.0230

Fig 16