```
function motorReadyNEGS(subject, block, comb)

% ------------------------------------------------------------------
% Motor readiness experiment
% Belopolsky vs. Lawrence
% Written by: Michael Puntiroli, November 2014
%             University of Geneva
%-------------------------------------------------------------------
    disp('00');
try
    disp('0');
    trl.exp = 1;
    trl.sub = str2num(subject);
    trl.blk = str2num(block);
    trl.comb = str2num(comb);
    disp('01');
%     trl.sub = subject;
%     trl.blk = block;
%     trl.comb = comb;
    eyeTrackerFlag = input('Eyelink (0/1)?: ');
    % set to 1 to do eye tracker recording
    % set to 0 if you want to test the experiment without eye tracker

    %--------------------------
    % initialize
    %--------------------------
    disp('02');
    [gfx trl] = initScreen(trl);
    % SUBFUNCTION BELOW
    disp('03');
    [gfx] = initGFX(gfx);
    % SUBFUNTION IN SEPARATE M-FILE
    disp('04');
    [gfx trl] = initSub(gfx, trl);
    % SUBFUNCTION BELOW
    if trl.overwrite_decision == 2,
        quitExp;
        % SUBFUNCTION BELOW
        fprintf('\nCANCELLED!\n');
        return;
    end
    disp('05');
    fid = fopen(trl.log_name,'w');
    fclose(fid);
    disp('06');
    ShowHideWinTaskbarMex (0);
    save(trl.gfx_name, 'gfx');
    % save the gfx structure once per block
    disp('07');
    % INIT EYELINK
    if eyeTrackerFlag == 1,
        el = exp_el_init(gfx, trl);
    else
        el = NaN;
    end
    disp('08');
    % CALIBRATE EYELINK
    if eyeTrackerFlag == 1,
```

```
        result=EyelinkDoTrackerSetup(el);
        if result==el.TERMINATE_KEY
            return;
        end;
        Eyelink('message', 'Block_Start');
        Eyelink('WaitForModeReady', 500);
    end

    disp('09');
    %--------------------------
    % show instruction
    %--------------------------
    dispInstruction(gfx, trl);
    % SUBFUNCTION BELOW

    %--------------------------
    % show stimuli
    %--------------------------

    % start of the trial loop:
    for i = 1:trl.n_trial,

        % show blank screen while preparing/drawing
        Screen('FillRect',trl.mainWnd,gfx.bColor);
        Screen('Flip', trl.mainWnd);

        %-----------------------------------
        % prepare trial variables
        %-----------------------------------

        trl.num = i;
        [trl] = initTrial(gfx, trl);
        % SUBFUNCTION BELOW
        WaitSecs(1); % minimum intertrial interval: 1000 ms

        % start eye tracker recording
        if eyeTrackerFlag == 1,
            exp_el_start(el, trl, gfx.cx, gfx.cy); % subfunction stored
in separate m-file
        end

        %--------------------------
        % show fixation display
        %--------------------------

        for counter=1:trl.fixFrms, % fixation frames

            % fixation cross
            Screen('FillRect', trl.mainWnd, gfx.fColor, gfx.fixXY);

            if trl.lineIdx == 1,
                 % saccade target line
                Screen('DrawLines', trl.mainWnd, trl.lineXY, gfx.linePen,
gfx.fColor);
            end

            % semi colored circles
```

```
            Screen('FrameOval', trl.mainWnd, trl.fixCols, gfx.OvalXY,
gfx.cirPen); % penWidth is 5

            % test circle: only to check if the circles are placed
correctly
            % Screen('FrameOval',trl.mainWnd,gfx.fColor, gfx.OvalTest,
2);

            %gray placeholders
            Screen('FrameRect', trl.mainWnd, gfx.fColor, gfx.masks,
gfx.penWfigs);

            Screen('Flip', trl.mainWnd);

            if counter == 1,
                st_fix = GetSecs;
            end

            if eyeTrackerFlag == 1 && counter == 1,
                Eyelink('Message', ['Fix on']);
            end
        end

%         WaitSecs(0.2);
%         KbWait;

        %-----------------------------------
        % first stimulus display
        %-----------------------------------

        if trl.SOA < 0 && trl.SOA > - 0.090, % for SOAs < 0, show DT
first

            for counter=1:trl.SOAFrms, % this only works if negative SOAs
< DT duration

                % fixation cross
                Screen('FillRect', trl.mainWnd, gfx.fColor, gfx.fixXY);

                if trl.lineIdx == 1,
                     % saccade target line
                    Screen('DrawLines', trl.mainWnd, trl.lineXY,
gfx.linePen, gfx.fColor);
                end

                % semi colored circles
                Screen('FrameOval', trl.mainWnd, trl.fixCols, gfx.OvalXY,
gfx.cirPen); % penWidth is 5

                % discrimination target display
                Screen('DrawLines', trl.mainWnd, trl.crosses,
gfx.penWfigs, gfx.fColor);

                Screen('Flip', trl.mainWnd);

                if counter == 1,
                    en_fix = GetSecs;
                    trl.fixDur = en_fix - st_fix;
```

```
                    st_DT = en_fix;
                end

                if eyeTrackerFlag == 1 && counter == 1,
                    Eyelink('Message', ['DT on']);
                end
            end


        elseif trl.SOA < 0 && trl.SOA < - 0.090,

            for counter=1:gfx.DTFrms, % display frames
                % fixation cross
                Screen('FillRect', trl.mainWnd, gfx.fColor, gfx.fixXY);

                if trl.lineIdx == 1,
                     % saccade target line
                    Screen('DrawLines', trl.mainWnd, trl.lineXY,
gfx.linePen, gfx.fColor);
                end

                % semi colored circles
                Screen('FrameOval', trl.mainWnd, trl.fixCols, gfx.OvalXY,
gfx.cirPen); % penWidth is 5

                % discrimination target display
                Screen('DrawLines', trl.mainWnd, trl.crosses,
gfx.penWfigs, gfx.fColor);

                Screen('Flip', trl.mainWnd);

                if counter == 1,
                    en_fix = GetSecs;
                    trl.fixDur = en_fix - st_fix;
                    st_DT = en_fix;
                end

                if eyeTrackerFlag == 1 && counter == 1,
                    Eyelink('Message', ['DT on']);
                end
            end


        elseif trl.SOA == 0, %if SOA == 0, show ST and DT

            for counter=1:gfx.DTFrms, % display frames

                % fixation cross
                Screen('FillRect', trl.mainWnd, gfx.fColor, gfx.fixXY);

                if trl.lineIdx == 1,
                     % saccade target line
                    Screen('DrawLines', trl.mainWnd, trl.lineXY,
gfx.linePen, gfx.fColor);
                end

                % fully colored circles (including ST)
```

```
                Screen('FrameOval', trl.mainWnd, trl.STCols, gfx.OvalXY,
gfx.cirPen); % penWidth is 5

                % discrimination target display
                Screen('DrawLines', trl.mainWnd, trl.crosses,
gfx.penWfigs, gfx.fColor);

                Screen('Flip', trl.mainWnd);

                if counter == 1,
                    en_fix = GetSecs;
                    trl.fixDur = en_fix - st_fix;
                    st_ST = en_fix;
                    st_DT = en_fix;
                end

                if eyeTrackerFlag == 1 && counter == 1,
                    Eyelink('Message', ['ST on']);
                end
            end


        elseif trl.SOA > 0, %if SOA > 0, show ST first

            for counter=1:trl.SOAFrms,

                % fixation cross
                Screen('FillRect', trl.mainWnd, gfx.fColor, gfx.fixXY);

                if trl.lineIdx == 1,
                     % saccade target line
                    Screen('DrawLines', trl.mainWnd, trl.lineXY,
gfx.linePen, gfx.fColor);
                end

                % fully colored circles (including ST)
                Screen('FrameOval', trl.mainWnd, trl.STCols, gfx.OvalXY,
gfx.cirPen); % penWidth is 5

                %gray placeholders
                Screen('FrameRect', trl.mainWnd, gfx.fColor, gfx.masks,
gfx.penWfigs);

                Screen('Flip', trl.mainWnd);

                if counter == 1,
                    en_fix = GetSecs;
                    trl.fixDur = en_fix - st_fix;
                    st_ST = en_fix;
                end

                if eyeTrackerFlag == 1 && counter == 1,
                    Eyelink('Message', ['ST on']);
                end
            end

        end
%         WaitSecs(0.2);
```

```
%         KbWait;

        %---------------------------------------
        % show second stimulus display
        %---------------------------------------

        if trl.SOA < 0 && trl.SOA > -0.090, % AND NEGATIVE SOA SMALLER
THAN DT

            for counter=1:(gfx.DTFrms - trl.SOAFrms), % remainder of the
DT duration

                % fixation cross
                Screen('FillRect', trl.mainWnd, gfx.fColor, gfx.fixXY);

                if trl.lineIdx == 1,
                     % saccade target line
                    Screen('DrawLines', trl.mainWnd, trl.lineXY,
gfx.linePen, gfx.fColor);
                end

                % fully colored circles (including ST)
                Screen('FrameOval', trl.mainWnd, trl.STCols, gfx.OvalXY,
gfx.cirPen); % penWidth is 5

                % discrimination target display
                Screen('DrawLines', trl.mainWnd, trl.crosses,
gfx.penWfigs, gfx.fColor);

                Screen('Flip', trl.mainWnd);

                if counter == 1,
                    st_ST = GetSecs;
                end

                if eyeTrackerFlag == 1 && counter == 1,
                    Eyelink('Message', ['ST on']);
                end

            end

        elseif trl.SOA == 0,
            % do not show anything
        elseif trl.SOA > 0,

            for counter=1:gfx.DTFrms, % show entire DT duration

                % fixation cross
                Screen('FillRect', trl.mainWnd, gfx.fColor, gfx.fixXY);

                if trl.lineIdx == 1,
                     % saccade target line
                    Screen('DrawLines', trl.mainWnd, trl.lineXY,
gfx.linePen, gfx.fColor);
                end

                % fully colored circles (including ST)
```

```
                Screen('FrameOval', trl.mainWnd, trl.STCols, gfx.OvalXY,
gfx.cirPen); % penWidth is 5

                % discrimination target display
                Screen('DrawLines', trl.mainWnd, trl.crosses,
gfx.penWfigs, gfx.fColor);

                Screen('Flip', trl.mainWnd);

                if counter == 1,
                    st_DT = GetSecs;
                end

                if eyeTrackerFlag == 1 && counter == 1,
                    Eyelink('Message', ['DT on']);
                end

            end

          elseif trl.SOA < 0 && trl.SOA < -0.090,

            for counter=1:abs(gfx.DTFrms - trl.SOAFrms), % to decide how
long to show the mask
                                                         % for before ST

                % fixation cross
                Screen('FillRect', trl.mainWnd, gfx.fColor, gfx.fixXY);

                    if trl.lineIdx == 1,
                     % saccade target line
                    Screen('DrawLines', trl.mainWnd, trl.lineXY,
gfx.linePen, gfx.fColor);
                    end

                % semi colored circles
                Screen('FrameOval', trl.mainWnd, trl.fixCols, gfx.OvalXY,
gfx.cirPen); % penWidth is 5

                %gray placeholders
                Screen('FrameRect', trl.mainWnd, gfx.fColor, gfx.masks,
gfx.penWfigs);

                Screen('Flip', trl.mainWnd);

                if counter == 1,
                    en_DT = GetSecs;
                    trl.DTDur = en_DT - st_DT;
                end

                if eyeTrackerFlag == 1 && counter == 1,
                Eyelink('Message', ['MS on']);
                end

            end


        end
%         WaitSecs(0.2);
```

```
%         KbWait;



        %---------------------------------------
        % show mask display
        %---------------------------------------


        for counter=1:gfx.mskFrms, %

            % fixation cross
            Screen('FillRect', trl.mainWnd, gfx.fColor, gfx.fixXY);

            if trl.lineIdx == 1,
                 % saccade target line
                Screen('DrawLines', trl.mainWnd, trl.lineXY, gfx.linePen,
gfx.fColor);
            end

            % fully colored circles (including ST)
            Screen('FrameOval', trl.mainWnd, trl.STCols, gfx.OvalXY,
gfx.cirPen); % penWidth is 5

            %gray placeholders
            Screen('FrameRect', trl.mainWnd, gfx.fColor, gfx.masks,
gfx.penWfigs);

            Screen('Flip', trl.mainWnd);

            if trl.SOA < -0.090 && counter == 1,
                st_ST = GetSecs;
                trl.ST2DT = st_DT - st_ST;
            end

            if trl.SOA > - 0.090 && counter == 1,
                en_DT = GetSecs;
                trl.ST2DT = st_DT - st_ST;
                trl.DTDur = en_DT - st_DT;
            end

            if trl.SOA > -0.090 && eyeTrackerFlag == 1 && counter == 1,
                Eyelink('Message', ['MS on']);
            end

            if trl.SOA < -0.090 && eyeTrackerFlag == 1 && counter == 1,
                Eyelink('Message', ['ST on']);
            end
        end

%         WaitSecs(0.2);
%         KbWait;

        %---------------------------------------------------------------
        % offline analysis of the eye movement behaviour in the trial
        %---------------------------------------------------------------

        if eyeTrackerFlag == 1,
```

```
            Eyelink('StopRecording');
            dat_name = [trl.dir_name, 'b', int2str(trl.blk), 't',
int2str(trl.num), '.dat'];
            playbackResult(trl.num) = EL2_playback(sprintf(dat_name));
            for c = 1:5,
                if exist(dat_name) > 0,
                    [eye] = resetEye;
                    data = textread(dat_name);
                    [eye] = scalculate(gfx,trl,eye,data);
                    break;
                elseif c == 5, % resetting all the eye structure if there
was a problem with the mexfile ...
                    [eye] = resetEye;
                else
                    WaitSecs(0.1);
                end
            end
        else % if there was no eye tracker recording, reset all anyway
...
            [eye] = resetEye;
        end

        %---------------------------------------
        % show error and response display
        %---------------------------------------

        if eye.error > 0, % if there was an error (e.g., break of
fixation, slow response ...) detected,
            % show a written feedback message on screen
            strInfo = eye.err_str;

DrawFormattedText(trl.mainWnd,strInfo,'center',gfx.cy,gfx.fColor,150,[],[
],5);
            Screen('Flip', trl.mainWnd);
            WaitSecs(1);
        end

        strInfo = ['ou'];

DrawFormattedText(trl.mainWnd,strInfo,'center',gfx.cy,gfx.fColor,150,[],[
],5);
        Screen('DrawLines', trl.mainWnd, trl.resp_figs, gfx.penWfigs,
gfx.fColor);
        Screen('Flip', trl.mainWnd);

        % get response
        keyIsDown = 0;
        initSecs = GetSecs;
        while keyIsDown == 0
            [keyIsDown, secs, keyCode, deltaSecs] = KbCheck;
        end
        endSecs = GetSecs;
        trl.RT = endSecs - initSecs;

        resp = find(keyCode == 1);

        if strcmp(KbName(resp(1)), 'q'), % press q to exit experiment
manually
```

```
            quitExp;
            fprintf('\nCANCELLED!\n');
            if eyeTrackerFlag == 1,
                exp_el_exit(trl);
            end
            return;
        end

         if strcmp(KbName(resp(1)), 'u'), % press q to exit experiment
manually
            if eyeTrackerFlag == 1,
                result=EyelinkDoTrackerSetup(el);
                if result==el.TERMINATE_KEY
                    return;
                end;
                Eyelink('WaitForModeReady', 500);
            end
        end


        % this part is related to the staircase procedure:
        % determine trl.prev for the current trial by looking at trl.prev
        % of the last trial
        if trl.num == 1,
            trl.prev = -999; % there is no previous trial
        elseif trl.prev == 1, % to ensure the 2 down, one up rule:
otherwise the staircase would go down for every consecutive right answer
            trl.prev = 0;
        else
            trl.prev = trl.correct; % before we assign trl.correct of the
current trial we put trl.correct of the previous trial in trl.prev
        end


        % which key has been pressed?
        if strcmp(KbName(resp(1)), 'y'), % left
            trl.resp = 1;
        elseif strcmp(KbName(resp(1)), 'm'), % right
            trl.resp = 2;
        else
            trl.resp = 0;
        end

        if trl.resp == trl.IdDT,
            trl.correct = 1;
        elseif trl.resp == 0, % if a completely different key has been
pressed, show an error message on screen
            trl.correct = -999;
            beep;
            DrawFormattedText(trl.mainWnd,'Fausse réponse! Appuyer sur
''y'' ou ''m''!','center',gfx.cy+130,gfx.fColor,150,[],[],5);
            Screen('Flip',trl.mainWnd);
            WaitSecs(1);
        else
            trl.correct = 0;
            beep;
        end
```

```
%         dat_name   =
[trl.dir_name,'b',block,'t',int2str(trl.num),'.mat'];
        dat_name   =
[trl.dir_name,'b',int2str(trl.blk),'t',int2str(trl.num),'.mat'];
        save(dat_name, 'trl', 'eye');
        % save the trl and eye structures into a mat-file ...

      % write the most important variables in the logfile
        fid = fopen(trl.log_name,'a');
                fprintf(fid, '%d %d %d %d %d ',trl.exp, trl.sub, trl.blk,
trl.comb, trl.num);
                fprintf(fid, '%d %d %d %d %d %d ',trl.NpST, trl.STCol,
trl.pSTCol, trl.attCol, trl.SOA*1000, trl.lineIdx);
                fprintf(fid, '%d %d %d %d %d %d ',round(trl.fixDur*1000),
eye.fixDurEL, round(trl.ST2DT*1000), eye.SOAEL, round(trl.DTDur*1000),
eye.DTDurEL);
                fprintf(fid, '%d %d %d %d %d %d ',trl.arrIdx, trl.STLoc,
trl.STLocClock, trl.DTLoc, trl.DTLocClock, round(trl.RT*1000));
                fprintf(fid, '%d %d %d %d ',trl.IdDT, trl.resp,
trl.correct, trl.prev);
                fprintf(fid, '%.2f %.2f %.2f %.2f %.2f ',eye.stx,
eye.sty, eye.enx, eye.eny, eye.amp);
                fprintf(fid, '%d %d %d %d %d %d\n', eye.STon, eye.stt,
eye.ent, eye.SRT, eye.SaccMaskon, eye.error);
        fclose(fid);


    end % this is then end of the trial loop


    %--------------------------
    % show end of exp
    %--------------------------

    ListenChar(0);

    dispEndInfo(gfx,trl); % Thank you screen

    save main; % save ALL variables into main.mat (this will be
overwritten ever time; it's more for debugging ...

    % "release" the eye tracker
    if eyeTrackerFlag == 1,
        exp_el_exit(trl);
    end

    WaitSecs(0.2);
    KbWait;

    %clearing things
    quitExp;


catch

    %this "catch" section executes in case of an error in the "try"
section
    %above.  Importantly, it closes the onscreen window if its open.
```

```
    quitExp;
    if eyeTrackerFlag == 1,
        exp_el_exit(trl);
    end
    psychrethrow(psychlasterror);

end % end of main function




%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%
%-------------------------------------------------------------------------
--
% SUBFUNCTIONS
%-------------------------------------------------------------------------
--
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%

% ------------------------------------------------
% initialization: screen and other system stuff
% ------------------------------------------------
function [gfx trl] = initScreen(trl)

    HideCursor;
    AssertOpenGL; %Check if PTB-3 is properly installed on the system

    Screen('Preference', 'SkipSyncTests', 0);
    %Screen('Preference', 'SkipSyncTests', 1); % use this only for
testing with a TFT monitor!!!
    %Get the list of screens and choose the one with the highest screen
number.
    screens=Screen('Screens');
    screenNumber=max(screens);


    % Open a double buffered fullscreen window
    [trl.mainWnd wsize] = Screen('OpenWindow',screenNumber); % Open On
Screen window, trl.mainWnd
    gfx.cx = wsize(3)/2; %center x position
    gfx.cy = wsize(4)/2; %center y position
    gfx.h_pixel = wsize(3);
    gfx.v_pixel = wsize(4);

    gfx.ifi=Screen('GetFlipInterval', trl.mainWnd); % inter-frame
interval (how many milliseconds until the next refresh)

    % configure text
    Screen('TextFont',trl.mainWnd,'Arial');
    Screen('TextSize',trl.mainWnd,18);

    ListenChar(2);

    % initialize random number generator
    rand('state', sum(100*clock));
```

```
end


% ---------------------------------------------
% initialization: subject and block parameters
% ---------------------------------------------
function [gfx trl] = initSub(gfx, trl)

    % -------------------------------------
    % set up subject file and directory
    % -------------------------------------

    % create the data folder (if it doesn't exist yet)
    if exist('data') == 0,
        mkdir('data');
    end

    % create the subject folder (if it doesn't exist yet)
    trl.dir_name = ['./data/e' int2str(trl.exp) 's', int2str(trl.sub)
'/'];
    if exist(trl.dir_name) == 0,
        mkdir(trl.dir_name);
    end

    % create strings to name the logfile and the GFX-file for the current
subject
    % and block
    trl.log_name   =
[trl.dir_name,'s',int2str(trl.sub),'b',int2str(trl.blk),'.log'];
    trl.gfx_name   =
[trl.dir_name,'s',int2str(trl.sub),'b',int2str(trl.blk),'-GFX','.mat'];
    % check if it already exists (danger of overwriting data!!!!)
    % if it does, ask if it should be overwritten give the logname back
empty and exit the experiment
    trl.overwrite_decision = 0;
    if exist(trl.log_name) > 0 & trl.sub ~= 99, % 99 is for testing
      Screen('FillRect',trl.mainWnd,gfx.bColor);
        WarningMsg = ['File name ', trl.log_name,' already exists !!!\n\n
Overwrite (y/n)?'];
        DrawFormattedText(trl.mainWnd,
WarningMsg,'center','center',gfx.fColor,150);
        Screen('Flip',trl.mainWnd);
      WaitSecs(0.2);
        while trl.overwrite_decision == 0,
          keyIsDown = 0;
            while keyIsDown == 0
                [keyIsDown, secs, keyCode, deltaSecs] = KbCheck;
            end
            resp = find(keyCode == 1);
            if strcmp(KbName(resp(1)), 'n'),
                trl.overwrite_decision = 2;
            elseif strcmp(KbName(resp(1)), 'y'),
                trl.overwrite_decision = 1;
            end
        end
    end
```

```
    trl.edf_name = ['s' num2str(trl.sub) 'b' num2str(trl.blk) '.edf'];
    % has to be shorter than 9 characters,
    % otherwise Eyelink crashes with an incomprehensible error
message!!!!!

    % -------------------------------------
    % set up subject and block parameters
    % -------------------------------------

    % get the assignments for the current block for all blocked variables
    subdesign = textread('subdesign.log');
    % design matrix columns:
    Clm.sub =      1; % subject
    Clm.comb =     2; % index of condition combinations for the current
block
    Clm.NpST =     3; % Number of possible saccade targets (2 or 6)
    Clm.STCol =    4; % saccade target ...
    Clm.pSTCol =   5; % ... possible saccade target ...
    Clm.attCol =   6; % ... and attended items color
    Clm.arrIdx =   7; % index for the aray of the possible saccade target
locations (right,down,left,up)

    idx = find(subdesign(:,Clm.sub) == trl.sub & subdesign(:,Clm.comb) ==
trl.comb);
    blockdesign = subdesign(idx,:);

    % determine colors for the current subject
    trl.STCol = blockdesign(Clm.STCol);
    trl.pSTCol = blockdesign(Clm.pSTCol);
    trl.attCol = blockdesign(Clm.attCol);
    trl.Colors = gfx.Colors(:,[trl.STCol,trl.pSTCol,trl.attCol]);

    % determine number of possible targets and assign colors
    trl.NpST = blockdesign(Clm.NpST);
    trl.arrCols = gfx.twelveGray;
    if trl.NpST == 2,
        trl.arrCols(:,[2,4]) = repmat(trl.Colors(:,2),1,2); % possible ST
color
        trl.arrCols(:,[8,10]) = repmat(trl.Colors(:,3),1,2); % attended
color
    elseif trl.NpST == 6,
        trl.arrCols(:,[1:6]) = repmat(trl.Colors(:,2),1,6);
        trl.arrCols(:,[7:12]) = repmat(trl.Colors(:,3),1,6);
    end

    trl.arrIdx = blockdesign(Clm.arrIdx);

    % randomization for all variables variable within-block
    % Create a design matrix: two columns
    STLoc = trl.NpST; % two or six possible locations
    DTLoc = 4; % four possible DT locations
    SOA   = 6; % six possible SOAs: [-100 -50 0 50 100 150]
    if trl.NpST == 2,
      rep = 3;
    elseif trl.NpST == 6,
      rep = 1;
    end % one or three repetitions for 144 trials
```

```
    design = fullfact([STLoc DTLoc SOA rep]);
    trl.n_trial = length(design(:,1));
    % randomize
    idx = randperm(trl.n_trial);
    design = design(idx,:);

    if trl.NpST == 6,
        DTLocs = repmat(1:12,1,12)';
        idx = randperm(144)';
        DTLocs = DTLocs(idx);
        design(:,2) = DTLocs;
    end


    % assign the first trials as "line trials"
    lineClm = zeros(trl.n_trial,1);
    lineClm(1:round(trl.n_trial/6)) = 1;    % Changed from 10 to 6
    design = [design, lineClm];

    % randomize again
    idx = randperm(trl.n_trial);
    trl.design = design(idx,:);


end


% ------------------------------------------------
% quit the experiment
% ------------------------------------------------
function quitExp
    Screen('CloseAll');
    ShowCursor;
    Priority(0);
    ListenChar(0);
end


% ------------------------------------------------
% initialization: trial parameters
% ------------------------------------------------

function [trl] = initTrial(gfx, trl)

    % to randomize the fixation duration
    trl.rand_int = rand*gfx.rand_int;
    trl.randFrms = round(gfx.rand_int/gfx.ifi); % MIGHT HAVE TO BE
CHANGED
    trl.fixFrms = gfx.fixFrms + trl.randFrms;

    % assign the saccade target
    trl.STLocIdx = trl.design(trl.num, 1);
    if trl.NpST == 2,
        STLocs = [2 4];
        trl.STLoc = STLocs(trl.STLocIdx);
    elseif trl.NpST == 6,
        trl.STLoc = trl.STLocIdx;
```

```
    end

    trl.DTLocIdx = trl.design(trl.num, 2);
    if trl.NpST == 2,
        DTLocs = [2 4 8 10];
        trl.DTLoc = DTLocs(trl.DTLocIdx);
    elseif trl.NpST == 6,
       trl.DTLoc = trl.DTLocIdx;
    end

    trl.fixCols = trl.arrCols;
    trl.STCols = trl.arrCols;
    trl.STCols(:,trl.STLoc) = trl.Colors(:,1); % saccade target color

    % shift the color vectors according to where the saccade target
arrray
    % is supposed to be (right,down,left,up)
    trl.stimLocs = 1:12; % to get the position of the ST in 'o'clock'
    if trl.arrIdx == 2,
        trl.fixCols = circshift(trl.fixCols,[0 3]);
        trl.STCols = circshift(trl.STCols,[0 3]);
        trl.stimLocs = circshift(trl.stimLocs,[0 -3]);
    elseif trl.arrIdx == 3,
        trl.fixCols = circshift(trl.fixCols,[0 6]);
        trl.STCols = circshift(trl.STCols,[0 6]);
        trl.stimLocs = circshift(trl.stimLocs,[0 -6]);
    elseif trl.arrIdx == 4,
        trl.fixCols = circshift(trl.fixCols,[0 9]);
        trl.STCols = circshift(trl.STCols,[0 9]);
        trl.stimLocs = circshift(trl.stimLocs,[0 -9]);
    end
    trl.STLocClock = trl.stimLocs(trl.STLoc);
    trl.DTLocClock = trl.stimLocs(trl.DTLoc);

    trl.lineIdx = trl.design(trl.num,end);
    trl.lineXY = [gfx.cx, gfx.Line_xs(trl.STLocClock); gfx.cy,
gfx.Line_ys(trl.STLocClock)];

    IdDT = randperm(2);
    trl.IdDT = IdDT(1); % assign a DT identity (1 = leftward, 2 =
rightward)

    % determine the shift for the line: staircase procedure (2 down, 1
up)
    if trl.num == 1,
        trl.shift = gfx.InitShift;
    else
        if trl.resp > 0 && trl.correct == 0, % if one of the two buttons,
but answer incorrect
            if trl.shift < gfx.blength, % if current shift smaller than
maximum
                trl.shift = trl.shift + gfx.shiftStep; % increase shift
            end
        elseif trl.correct == 1 && trl.prev == 1, % if last two answers
correct
            trl.shift = trl.shift - gfx.shiftStep;
            if trl.shift <= gfx.minShift,
                trl.shift = gfx.minShift;
```

```
            end
        end
    end

    % place the different figures in the circles
    trl.crosses = [];
    for f = 1:length(gfx.xs),
        curFig = gfx.cross; % if it is a target or distractor location
        if f == trl.DTLocClock,
            if trl.IdDT == 1,
                curFig(1,1:2) = -trl.shift*gfx.h_deg2pix;
            elseif trl.IdDT == 2,
                curFig(1,1:2) = trl.shift*gfx.h_deg2pix;
            end
        end
        curFig(1,:) = curFig(1,:)+gfx.xs(f); % x coordinates
        curFig(2,:) = curFig(2,:)+gfx.ys(f); % y coordinates
        trl.crosses = [trl.crosses,curFig];
    end

     % prepare the figures for the response display
     resp_figL = gfx.cross;
     resp_figL(1,1:2) = -trl.shift*gfx.h_deg2pix; % add left shift
     resp_figL(1,:) = resp_figL(1,:)+gfx.cx-gfx.h_ecc; % place to the
left of fixation
     resp_figL(2,:) = resp_figL(2,:)+gfx.cy; % place on the horizontal
meridian
     resp_figR = gfx.cross;
     resp_figR(1,1:2) = trl.shift*gfx.h_deg2pix; % add right shift
     resp_figR(1,:) = resp_figR(1,:)+gfx.cx+gfx.h_ecc; % place to the
left of fixation
     resp_figR(2,:) = resp_figR(2,:)+gfx.cy; % place on the horizontal
meridian
     trl.resp_figs = [resp_figL, resp_figR];

    trl.SOA = gfx.SOAs(trl.design(trl.num, 3));
    trl.SOAFrms = gfx.SOAFrms(trl.design(trl.num, 3));

end


% ----------------------------------------
% reset all entries in the eye structure
% ----------------------------------------
function [eye] = resetEye
      eye.fixDurEL = 0;
      eye.STon = 0;
      eye.DTon = 0;
      eye.DTDurEL = 0;
      eye.SOAEL = 0;
      eye.Maskon = 0;
      eye.stx = -999;
      eye.sty = -999;
      eye.enx = -999;
      eye.eny = -999;
      eye.amp = -999;
      eye.stt = -999;
      eye.ent = -999;
```

```
      eye.ONS = -999;
      eye.OFS = -999;
      eye.SRT = -999;
      eyel.SaccDur = 0;
      eye.error = 0;
      eye.err_str = [];
    eye.SaccMaskon = -999;
end


% ---------------------
% instruction display
% ---------------------
function dispInstruction(gfx,trl)

    strInfo1 = ['Regardez le plus vite possible vers le cercle coloré.'];

    Screen('FillRect',trl.mainWnd,gfx.bColor);
    DrawFormattedText(trl.mainWnd,strInfo1,'center',gfx.cy-
240,gfx.fColor,150,[],[],5);

    strInfo2 = ['Indiquez à la fin de l''essai si'];
    DrawFormattedText(trl.mainWnd,strInfo2,'center',gfx.cy-
120,gfx.fColor,150,[],[],5);

    Screen('DrawLines', trl.mainWnd, gfx.resp_figs, gfx.penWfigs,
gfx.fColor);
    strInfo3 = ['ou'];

DrawFormattedText(trl.mainWnd,strInfo3,'center',gfx.cy,gfx.fColor,150,[],
[],5);

    strInfo4 = ['était présenté dans un des cercles.'];

DrawFormattedText(trl.mainWnd,strInfo4,'center',gfx.cy+130,gfx.fColor,150
,[],[],5);

    strInfo5 = ['Appuyez sur un bouton pour commencer.'];

DrawFormattedText(trl.mainWnd,strInfo5,'center',gfx.cy+240,gfx.fColor,150
,[],[],5);
    Screen('Flip', trl.mainWnd);

    WaitSecs(0.2);
    KbWait;

end

% ---------------------
% end display
% ---------------------
function dispEndInfo(gfx,trl)
    Screen('FillRect',trl.mainWnd,gfx.bColor);
    blockTXT = ['Fin de bloc ', num2str(trl.blk), '! Merci!'];
    DrawFormattedText(trl.mainWnd,
blockTXT,'center','center',gfx.fColor,40);
    DrawFormattedText(trl.mainWnd,'Veuillez prévenir l''expérimentateur,
s.v.p.','center',gfx.cy+130,gfx.fColor,150,[],[],5);
```

```
    Screen('Flip',trl.mainWnd);
end

end
```