311603476
307963157

הקדמה

בעבודה זו נייצר ניסוי התנהגותי של חיפוש חזותי באמצעות מטלב, וננתח את הנתונים במטרה להראות את אפקטים קשביים המאפיינים סוג מטלות זה - **אפקט הפופ-אאוט**, לפיו בחיפוש חזותי של סט גירויים המאופיינים בתכונית אחת, מטרה הנבדלת בתכונית זו תאותר במהירות גבוהה וקבועה, ללא תלות בגודל סט הגירויים המוצג, ו**אפקט איחוד התכוניות**, לפיו בחיפוש חזותי של סט גירויים המאופיינים ביותר מתכונית אחת, מטרה הנבדלת בשילוב תכוניות אלו תאותר במהירות משתנה ויורדת, ביחס חיובי לגודל סט הגירוי (ככל שגודל שהסט גדול יותר, מהירות התגובה איטית יותר). נפנה כעת להסביר את השיטות בהם השתמשנו, נציג את הממצאים שהתקבלו ונדון במשמעויותיהן על בסיס תיאוריית איחוד התכוניות של אן טריזמן.

שיטות

באמצעות מטלב הכנו את ניסוי החיפוש החזותי. הניסוי נבנה בעזרת סקריפט ראשי בשם "Ex3_311603476_307963157" ופונקציה שכתבנו בשם display_stimuli, המייצרת את סט הגירויים החזותי בעבור כל צעד ניסויי.

ראשית, הגדרנו את חלון הניסוי שישאר קבוע עד תום הניסוי. את חלון זה ננקה בכל מעבר תמונה בניסוי. לחלון זה קראנו 'Visual Search Experiment', והסרנו ממנו את סרגלי הכלים למיניהם. לאחר מכן, קבענו משתנים בוליאנים (0 או 1) – run הקובע אם הניסוי רץ, trialBool הקובע אם צריך לחזור על הניסוי, stopExp הקובע אם צריך לעצור את הניסוי בעקבות מחסור בנתונים, וRun_Mode הקובע אם התוכנה פועלת בעצמה או שאכן יש נבדק המבצע את הניסוי.

```
% set boolean variables
run = 1;
trialBool = 0;
stopExp = 0;
run_mode = 1;

while run   % experiment big loop
    % for now, we hope the experiment will run only once
    run = 0;

    […]
```

כעת, השתמשנו בלולאת 'כל-עוד', שתנאי הכניסה שלה הוא שהניסוי רץ (Run = 1). במצב זה, הגדרנו פרמטרים – מספר החזרות על כל בלוק, גודל סט גירויים מינימלי, גודל סט גירויים מקסימלי והמרווח שבין הסטים השונים. יצרנו וקטור המכיל את גדלי הסטים ווקטור המכיל 0 ו1 המעיד על סוג החיפוש המתבצע.

```
    % to make sure each set size will appear once in each search type,
    % we combine the sizes vector with search type into a balanced 2-
dimension matrix
    [A B] = meshgrid(tempSymbols, tempConj);
    temp = reshape(cat(2,A',B'),[],2);
    % then randomize their order.
    randLocation = randperm(length(temp));
    RandomBlocks = temp(randLocation, :);
```

```
[…]
```

בשלב זה, על מנת לוודא שכל גודל סט יופיע פעם אחת עבור כל סוג חיפוש, איחדנו את וקטור הגדלים עם וקטור סוג החיפוש בצורה מאוזנת למערך דו מימדי באמצעות פונקציית meshgrid (שמחזירה בשני וקטורים את קומבינציית כל התנאים האפשריים) ופונקציית reshape (שמאחדת את שני הוקטורים למערך דו מימדי), ויצרנו פרמוטציה של סידור הבלוקים – כל בלוק הינו בעל גודל סט מסויים וסוג חיפוש מסויים. בנוסף, שמרנו בנפרד את מספר התנאים ואת מספר החזרות הכולל של כל הניסוי. לאחר מכן יצרנו וקטור באורך מספר החזרות לכל בלוק המכיל בחציו '0' ובחציו '1' אשר יעיד על הימצאות/אי-הימצאות המטרה.

```
    % Creating gaussian mixed distribution for each of the four conditions
    %       with four values each, representing four set sizes
    % Then, numbers equal to half of the trials, were taken randomly
    %       from the "Response" distribution so that the automated run will
    %       have "Realistic" response time.
    if run_mode
        gm = gmdistribution([0.6 0.6 0.6 0.6], [0.1 0.1 0.1 0.1]);
        AutoFTar = random(gm, nTrials/2);
        gm = gmdistribution([0.6 0.7 0.7 0.8], [0.1 0.1 0.1 0.1]);
        AutoFnoTar = random(gm, nTrials/2);
        gm = gmdistribution([1.0 1.4 1.7 2], [0.1 0.1 0.2 0.2]);
        AutoCTar = random(gm, nTrials/2);
        gm = gmdistribution([1.0 1.5 1.9 2.3], [0.1 0.1 0.2 0.2]);
        AutoCnoTar = random(gm, nTrials/2);
    end
```

בנוסף, הגדרנו את המשתנים עבור הריצה האוטומטית. באמצעות הפונקציות gmdistribution ו-random יצרנו 4 מערכים דו-מימדיים המייצגים את תנאי הניסוי השונים (יש/אין מטרה-סוג החיפוש), שכל אחד מהם מכיל 15 זמני תגובה בעבור כל גודל סט גירוי. הערכים מתקבלים מתוך התפלגות גיאוסיאנית בעלת ממוצעים וסטיות תקן קבועים (המתאימים לתוצאות הרצויות).

בשלב זה, התחלנו לאסוף את הנתונים. הגדרנו שיופיע טקסט לנבדקים, המסביר לנבדקים את מהלך הניסוי, והגדרנו שמעבר לחלון הבא יהיה אפשרי רק על ידי לחיצה על מקש הרווח. עם לחיצה על רווח, הגדרנו לולאה עבור כל בלוק. ערבבנו מחדש את סדר וקטור הימצאות המטרה, בכדי לנטרל אפקט סדר אפשרי, שיהווה קונפאונד לתוצאות הניסוי. הגדרנו משתנה tempData שיכיל זמני תגובה ודיוק עבור כל בלוק, ובחרנו את צורת הגירוי (x או o) באופן רנדומלי, והצגנו את ההוראות הספציפיות לבלוק הנוכחי. עבור הריצה האוטומטית, הוגדרו 4 וקטורי סדר שמיועדים להחליט על סדר הופעת זמני התגובה שנקבעו קודם לכן, ו4 משתני ספירה שמיועדים להתקדמות מותאמת של ארבעת וקטורי הסדר.

```
if run_mode
    FNT = randperm(nTrials/2);
    FNTcount = 1;
    FT = randperm(nTrials/2);
    FTcount = 1;
    CNT = randperm(nTrials/2);
    CNTcount = 1;
    CT = randperm(nTrials/2);
    CTcount = 1;
```

```
End
```

כעת, הגדרנו לולאת חזרות עבור כל בלוק. התחלנו ספירת זמן תגובה באמצעות הפונקציה tic, הצגנו את סט הגירוי הנוכחי באמצעות פונקציה שבנינו בשם display_stimuli (עליה יוסבר בהמשך). בשלב זה המערכת מחכה לתשובת הנבדק. במקרה של ריצה אוטומטית, עבור כל תנאי ניסוי (יש/אין מטרה-סוג החיפוש) התבצעה הפסקה באופן שונה (בעזרת פונקציית pause) בהתאם למערכי זמני התגובות אותם הגדרנו לפני כן, ובעזרת וקטורי הסדר ומשתני הספירה הרלוונטיים. הגדרנו משתנה הסתברותי בעזרת פונקציית rand, אשר נועד לסימון תשובה נכונה ב95% אחוז מהמקרים במהלך הריצה האוטומטית. לאחר קבלת תשובה מספקת (לחיצה על a או l), הקוד מפסיק להמתין לתשובה, מפסיק את מדידת זמן התגובה באמצעות פונקציית toc ובודק את הדיוק (תשובה נכונה או שגויה) עבור החזרה הנוכחית. את זמן התגובה והדיוק אנו מכניסים למשתנה tempData, מנקים את חלון הניסוי וחוזר חלילה על חזרות הניסוי עד גמר הבלוק.

```
for a = 1:nTrials
    tic;    % start counting response time
    % display current stimuli
    display_stimuli(Nsymbols(type), isTarget(a), isConj(type),correct_ans)
    if run_mode
        randAns = rand();
        if isConj(type)
            if isTarget(a)
                pause(AutoCTar(CT(CTcount),Nsymbols(type)/stepSize));

                CTcount = CTcount +1;
                if randAns > 0.05
                    key = 'a';
                else
                    key = 'l';
                end
            else
                […]
        else
            if isTarget(a)
               […]
            end
        end
    else
        pause();
        key = h.CurrentCharacter;
    end

         […]

    % temporarily saves the variables
    tempData = [tempData ; RT, acc];
    clf; % clear figure
end
if sum(tempData(:,2)) < 20    % if doesn't have enough data
    trialBool = 1;            % stop the experiment and start over
    break;
end
% save data in a structure called ExperimentData
% with blocks' numbers as sub-structures
ExperimentData.(sprintf('Block%d',type)).setSize = Nsymbols(type);
ExperimentData.(sprintf('Block%d',type)).targetOrder = isTarget;
ExperimentData.(sprintf('Block%d',type)).isConjunction = isConj(type);
ExperimentData.(sprintf('Block%d',type)).RT = tempData(:,1);
```

```
ExperimentData.(sprintf('Block%d',type)).accuracy = tempData(:,2);
end
```

בסיום כל בלוק, בדקנו שיש למעלה מ20 תשובות נכונות, ואם התגלה שלא, הניסוי נעצר והנבדק צופה בהודעה המציעה לו לחזור על הניסוי או לצאת ממנו לחלוטין (הקוד יוסבר בהמשך). בשלב זה, שמרנו את המידע במבנה נתונים הנקרא ExperimentData כך שעבור כל בלוק קיים תת-מבנה נתונים השומר את חמשת הנתונים הבאים – גודל הסט, וקטור הימצאות המטרה, סוג החיפוש, וקטור זמני התגובה ווקטור הדיוק. לאחר מכן, הלולאה ממשיכה לבלוק הבא. באמצעות פקודה save אנו שומרים את מבנה הנתונים תחת השם raw data.mat. בשלב זה פנינו לניתוח הנתונים.

בתחילה, בדקנו את המשתנה הבוליאני trialBool לוידוא שאנו אכן צריכים לבצע את ניתוח הנתונים, ואין צורך בחזרה מחודשת על הניסוי. לאחר מכן, נוקו הנתונים השייכים לתשובות השגויות והנתונים עבור זמני תגובה הגדולים מ3 שניות. לאחר הוצאת נתוני זמני התגובה, בדקנו שאכן נשארו לנו לפחות 20 תשובות עבור כל בלוק, כך שאם תנאי זה לא מתקיים, ניתוח הנתונים מופסק והנבדק צופה בהודעה המציעה לו לחזור על הניסוי או לצאת ממנו לחלוטין. כעת, חילקנו את הנתונים לשני חלקים, בהתאם לסוג החיפוש שהתבצע. עבור כל סוג חיפוש, חולקו הנתונים לשני חלקים, בהתאם לתנאי הימצאות/אי הימצאות המטרה. את ממוצעי זמני התגובה וסטיות התקן שמרנו במערכים דו מימדיים ובו זמנית אלו סודרו לפי גודל הסט.

```
% split data into two halves by first indicator: Feature or
% Conjunction
if isConj(n) == 0
    Feature.(sprintf('Block%d',n)) = ExperimentData.(sprintf('Block%d',n));
% now split this data into two halves by second indicator:
% Target or no-Target
helpVec = find(Feature.(sprintf('Block%d',n)).targetOrder == 0);
Feature.(sprintf('Block%d',n)).NoTarget =
                 Feature.(sprintf('Block%d',n)).RT(helpVec);
helpVec = find(Feature.(sprintf('Block%d',n)).targetOrder == 1);
Feature.(sprintf('Block%d',n)).Target =
                 Feature.(sprintf('Block%d',n)).RT(helpVec);
% in arrays, calculate mean response time and std of each
% block. Simultaneously, sort them by their set size.
FmeanRTnoTar(Feature.(sprintf('Block%d',n)).setSize/stepSize) =
                 mean(Feature.(sprintf('Block%d',n)).NoTarget);
FstdRTnoTar(Feature.(sprintf('Block%d',n)).setSize/stepSize) =
                 std(Feature.(sprintf('Block%d',n)).NoTarget);
FmeanRTTar(Feature.(sprintf('Block%d',n)).setSize/stepSize) =
                 mean(Feature.(sprintf('Block%d',n)).Target);
FstdRTTar(Feature.(sprintf('Block%d',n)).setSize/stepSize) =
                 std(Feature.(sprintf('Block%d',n)).Target);
else
[…]
end
```

```
    if trialBool
        g = text(0.5, 0.5, …
                 […]

        if run_mode
            pause(4);
            key = ' ';
```

```
        else
            pause();
            key = h.CurrentCharacter;
        end
    % wait for an adequate response
        while strcmpi(key, ' ') == 0 && strcmpi(key, '') == 0
            pause();
            key = h.CurrentCharacter;
        end

        if strcmpi(key, ' ') == 1      % if the participant wants to start
over
            run =1;                    % start over
        elseif strcmpi(key, '') == 1  % if the participant wants to quit
            stopExp = 1;               % quit
            break;
        end
        clf;    % clear figure
    end
end
close;
```

קטע הקוד הבא יתרחש בעבור trialBool = 1 (נדרשת חזרה נוספת על הניסוי) – תוצג הודעה לנבדק המאפשרת לו לחזור על הניסוי בלחיצת רווח, או לצאת בלחיצה על כפתור Esc. ניתן לראות הדגמה כיצד הריצה האוטומטית "מחקה" התנהגות אנושית ע"י המתנה למשך זמן קבוע מראש, ו"לחיצה" על מקש רווח, כך נעשה לאורך כל הניסוי (כשזמני ההמתנה משתנים לפי הצורך). בשלב זה סגרנו את לולאת ה'כל-עוד' שמכילה את כלל הניסוי וסגרנו את חלון הניסוי.

```
    coeffFeatTar = polyfit(tempSymbols, FmeanRTTar, 1);

    slopeValFeatTar = polyval(coeffFeatTar, tempSymbols);

    coeffFeatNoTar = polyfit(tempSymbols, FmeanRTnoTar, 1);

    slopeValFeatNoTar = polyval(coeffFeatNoTar, tempSymbols);

    coeffConjTar = polyfit(tempSymbols, CmeanRTTar, 1);

    slopeValConjTar = polyval(coeffConjTar, tempSymbols);

    coeffConjNoTar = polyfit(tempSymbols, CmeanRTnoTar, 1);

    slopeValConjNoTar = polyval(coeffConjNoTar, tempSymbols);

    [R_FeatTar, P_FeatTar] = corrcoef(tempSymbols, FmeanRTTar);

    [R_FeatNoTar, P_FeatNoTar] = corrcoef(tempSymbols, FmeanRTnoTar);

    [R_ConjTar, P_ConjTar] = corrcoef(tempSymbols, CmeanRTTar);

    [R_ConjNoTar, P_ConjNoTar] = corrcoef(tempSymbols, CmeanRTnoTar);
```

בעבור כל אחד מתנאי הניסוי, חישבנו את המקדמים והערכים המתאימים לפונקציה הליניארית בעזרת הפונקציות polyfit – המחזירה את המקדמים המתאימים לפולינום שצורתו היא המקורבת

ביותר לתוצאות שהתקבלו בניסוי, polyvalו – המחזירה את הערכים שנמצאים על הגרף בהתאם לציר הX. חישבנו את מתאם פרסון בעבור כל תנאי ניסוי בעזרת פונקציית corrcoef אשר מחזירה לנו את המתאם והp-value המתאים.

בשלב זה, פנינו להצגת הגרפים. הגרפים חולקו לשני חלונות – האחד עבור הימצאות המטרה והשני עבור היעדרותה. כל חלון מכיל ארבעה גרפים – גרף עבור כל ממוצעי זמני התגובה של כל סוג חיפוש ועוד גרף עבור הפונקציה הליניארית המותאמת.

<u>הפונקציה display_stimuli</u>

הפונקציה נבנתה למטרת הצגת גירוי בכל צעד בניסוי. זו פונקציה ללא משתני חזרה, רק תצוגה. משתני הקלט שלה הם – גודל הסט (4,8,12,16), האם יש מטרה (0,1), האם יש איחוד תכוניות (0,1) ומהי המטרה אותה אנו מחפשים (O או X).

הדבר הראשון אותו הפונקציה עושה הוא קביעת צבע המטרה, בעזרת פונקציית rand. לאחר מכן היא מסמנת לעצמה את הצורה המסיחה – בהתאם לקלט המטרה. כעת, על מנת לבחור מיקומים רנדומליים בעבור תצוגת הגירויים, הוגדר משתנה עם מספר שרירותי (50) שנבחר לייצג את כמות הפעמים בו אנו מחלקים את חלון תצוגת הניסוי. בשלב זה הגדרנו מערך דו-ממדי אשר כל שורה מייצגת גירוי וכל עמודה מייצגת קואורדינטות של אחד הצירים. הדבר האחרון שנעשה בטרם הפונקציה פונה לתצוגת הגירויים הוא יצירת וקטור תאים, אשר מכיל בכל תא את המסיח שנמצא קודם לכן.

הצגת הגירויים כוללת בתוכה התייחסות לארבע אופציות (ארבעת התנאים):

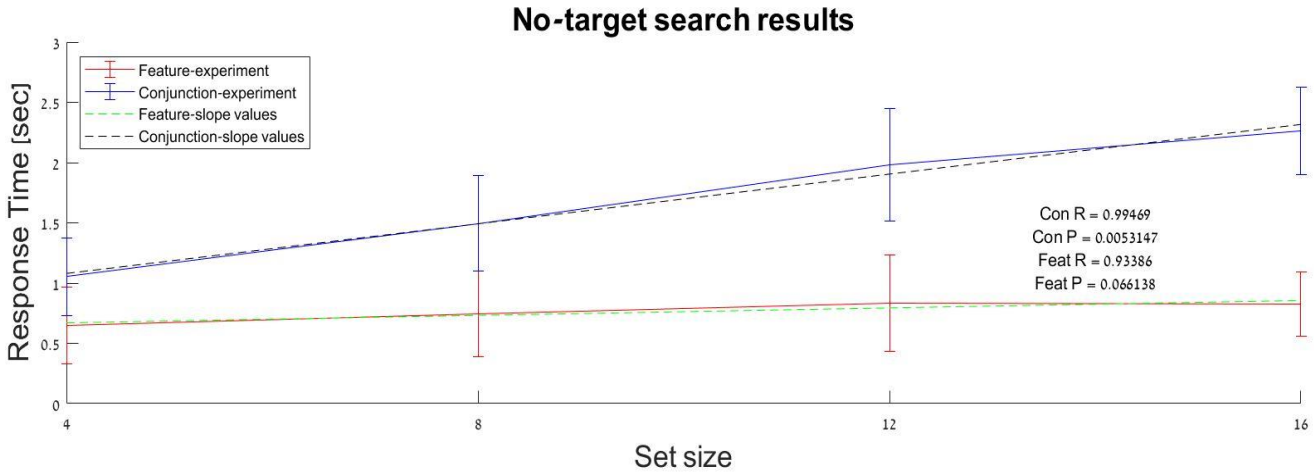
- תכונית בודדת כשיש מטרה – הפונקציה מוסיפה במיקום אקראי בוקטור תאים את הצורה אותה הנבדק צריך לחפש ומציגה את הוקטור הנ"ל לפי המיקומים שחושבו בטרם ההצגה. הצבע שבו יוצגו הגירויים הינו הצבע אשר נבחר בתחילת הפונקציה.
- תכונית בודדת כשאין מטרה – תצוגת וקטור התאים כפי שהוא.
- איחוד תכוניות – צבע הוצמד לכל צורת גירוי בהתאם לצבע הרנדומלי שהוגדר בהתחלה. לאחר מכן, חצי מאיברי וקטור התאים שונו כך שיכילו את הצורה אותה הנבדק צריך לחפש.
    - כשיש מטרה – נבחר באקראי אחד המיקומים של הצורות הנכונות והוא יהיה זה אשר יופיע בצבע של הצורות השגויות, כל השאר יופיעו לפי החלוקה הקודמת.
    - כשאין מטרה – הגירויים יופיעו בהתאם למה שנקבע עבורם קודם לכן, כל צורה במיקום רנדומלי ובצבע הקבוצתי שלה.

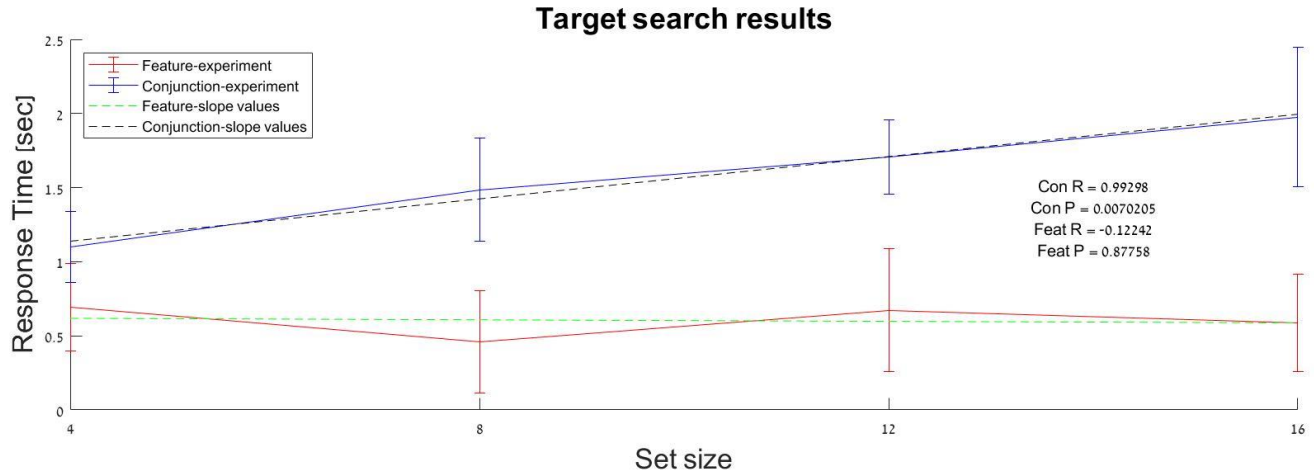
311603476
307963157

תוצאות ודיון

התקבלו 4 גרפים בעבור תנאי הימצאות המטרה ו4 גרפים בעבור תנאי היעדרות המטרה. מספר החזרות התקינות בכל בלוק מצויינות בנספח 1 (ראו מטה).

הגרפים מציגים את משך זמן התגובה הממוצע כפונקציה של גודל סט הגירויים בעבור סוג חיפוש איחוד תכוניות (בכחול), סוג חיפוש תכונית בודדת (באדום), והקוים הלינאריים המותאמים (בשחור ובירוק).





ניתן לראות כי הגרפים המתאימים לתכונית בודדת אינו מושפע מגודל סט הגירויים ($R = 0.918$ עבור תנאי היעדרות המטרה, $R = 0.102$ עבור תנאי הימצאות המטרה. שניהם בעלי $p > 0.05$). ממצא זה מדגים את אפקט הפופ-אאוט, שכן חוסר המובהקות של המתאם בתנאים אלו מדגימים שבתנאי בו יש מטרה הנבדלת בתכונית בודדת בלבד משאר הגירויים, הגירוי המטרה קופץ במהירות דומה להכרת האדם בכל תנאי הניסוי, ולכן זמן התגובה לא מושפע ממספר הגירויים המוצגים.

עם זאת, חשוב לשים לב שבעוד ששני המתאמים אינם מובהקים, המתאם בין גודל סט הגירויים לזמן התגובה בתנאי בו אין מטרה גבוה יותר משמעותית מהמתאם בין גודל סט הגירויים לזמן התגובה בתנאי בו יש מטרה. ממצא זה מוסבר בעקבות העובדה שבתנאי בו אין מטרה, התנהגות הנבדקים הינה מעבר רציף על כלל הגירויים המוצגים, לצורך וידוא שאכן אין מטרה מוצגת (בניגוד לתנאי בו

המטרה קיימת ו'קופצת לעין'). לכן, זמן התגובה אכן עולה בהתאם לכמות הגירויים. עם זאת, מכיוון שבתנאי זה כלל הגירויים הינם זהים, המעבר על כלל הגירויים ועיבודם הינו תהליך פשוט ומהיר יחסית, ולכן למרות ההבדל בין המתאמים, עדיין הממצא אינו מובהק.

בגרפים המתאימים לתנאי איחוד התכוניות נמצא קשר מובהק בין גודל סט הגירויים לזמן התגובה של הנבדק ($R = 0.996$ עבור תנאי הימצאות המטרה, $R = 0.998$ עבור תנאי היעדרות המטרה. שניהם בעלי $p < 0.005$). ממצא זה מדגים שבתנאים בהם גירוי המטרה נבדל משאר הגירויים בשילוב של יותר מתכונית אחת (כך שגירוי המטרה דומה לחלק מהגירויים בתכונית אחת ולחלק מהגירויים בתכונית שניה), זמן התגובה עולה משמעותית ככל שגודל סט הגירויים המוצג עולה. גם במקרה זה, בתנאי בו אין מטרה המתאם מעט גבוה יותר. בדומה לתנאי התכונית הבודדת, גם ממצא זה מצביע על כך שבתנאים בהם אין מטרה, הנבדק נדרש לעבור על כלל הגירויים המוצגים בטרם הנבדק מסיק את מסקנתו (ולא לעצור את החיפוש ברגע שהוא שם לב לגירוי המטרה). כמו כן, מכיוון שבמקרה זה מדובר בגירויים המורכבים משילוב קומבינצייה של 2 תכוניות בסיסיות שונות, זמן העיבוד עבור כל גירוי בפני עצמו גבוה יותר.

הסבר לממצאים ניתן לפי תיאוריית איחוד התכוניות של אן טריזמן. לפי תיאוריה זו, תהליך עיבוד של מרחב המכיל גירויים מתרחש **במקביל ובאופן אוטומטי**, כך שהתכוניות הבסיסיות (צבע, צורה, גודל, תנועה וכו') מחולצות באותו הזמן. לכן, כאשר גירוי המטרה נבדל בתכונית בודדת אחת משאר הגירויים המסיחים, מתקיים אפקט הפופ-אאוט – התכונית הבודדת של גירוי המטרה תעובד במקביל ובאופן אוטומטי, ללא צורך הפניית קשב, וכך הנבדק יכול לדווח על קיומו של גירוי המטרה באותו הזמן וללא תלות בגודל סט הגירויים (העיבוד המקבילי האוטומטי מנביע זאת).
לעומת זאת, בתנאי שילוב תכוניות, בו גירוי המטרה נבדל משאר הגירויים ע"י שילוב של שתי תכונות, אפקט הפופ-אאוט לא מתקיים, שכן בתנאי זה, לפי התיאוריה של אן טריזמן, אנו נדרשים לשלב כמה תכוניות כשייכות לאובייקט אחד. על מנת לבצע זאת, על הנבדק למקד את קשבו באיזורים ספציפיים בשדה החיפוש הויזואלי. לכן, על מנת למצוא את גירוי המטרה בתנאי זה, על הנבדק לעבור באופן שיטתי ומכוון על הגירויים השונים בכדי לבצע חיפוש סדרתי בין שילוב התכוניות המאפיינות את הגירויים השונים (כלומר, במקרה זה, מכיוון שהגירויים מורכבים מיותר מתכונית אחת, גם אם תהליך 'שליפת' התכוניות מתקיים באופן מקבילי ואוטומטי, תהליך עיבוד **המשמעות** הכוללת של מרחב הגירויים, כך שנוכל להעניק משמעות ולזהות את גירוי המטרה, כבר **אינו מקבילי ואינו אוטומטי**). ניתן לראות (הוסבר מעלה מדוע) שתוצאות ניסויינו תואמות לתיאורית איחוד התכוניות של טריזמן.

311603476
307963157

טבלת פירוט נתונים פר בלוק

מספר הבלוק	גודל סט	סוג חיפוש	ממוצע תגובה	מספר תשובות תקינות
1	4	תכונית	0.6715	29
2	16	איחוד תכוניות	2.121	26
3	8	איחוד תכוניות	1.49	28
4	12	איחוד תכוניות	1.848	29
5	4	איחוד תכוניות	1.077	28
6	16	תכונית	0.704	29
7	8	תכונית	0.605	27
8	12	תכונית	0.749	29

טבלת השוואת זמני תגובה לפי גודל סט, סוג חיפוש והימצאות/היעדרות מטרה

גודל בלוק	מטרה מוצגת		מטרה אינה מוצגת	
	תכונית	איחוד תכוניות	תכונית	איחוד תכוניות
4	0.694	1.098	0.65	1.053
8	0.458	1.486	0.741	1.495
12	0.672	1.706	0.833	1.98
16	0.589	1.977	0.827	2.265