**הסבר על הפרויקט:**

בחרתי כך שתהליך 0 יהיה ה-master הוא יקבל את הנתונים מהקובץ “input” יחלק את העבודה בין כל התהליכים כולל לעצמו וישלח את התוצאות של כל התהליכים לקובץ “output”.

בחרתי להחזיק את הנתונים של התמונות והנתונים של האובייקטים במערך תלת מימדי, מפני שהיה לי מאוד נוח להסתכל על זה ככה, לדוגמה: int\*\*\* picArr הוא מערך של כל התמונות יחד, אז: PicArr[0][2][5] = הערך של תמונה 0, בשורה 2, בעמודה 5.

**הסבר מפורט על דרך הביצוע:**מספר התהליכים חייב להיות קטן או שווה למספר התמונות על מנת שלא יהיו תהלכים שלא יעשו כלום.  
תהליך 0 מקבל את כל התמונות ואח"כ את כל האובייקטים ומחזיק אותם במערכים נפרדים, לאחר שהכל עבר בהצלחה, כל תהליך בודק אם מספר התמונות מתחלק בין התהליכים, אם כן מתחלק אז מספר התמונות לכל תהליך יוצא שווה בשווה בין כל התהליכים, אם לא מתחלק אז מספר התמונות לכל תהליך יוצא שווה בשווה ונותן את השארית לתהליך האחרון וכל תהליך שולח לתהליך 0 כמה תמונות הוא צריך לקבל.  
לאחר מכן, תהליך 0 שולח לכל התהליכים את כמות התמונות שכל תהליך צריך לעבוד עליו באמצעות “MPI\_Scatterv”.  
וכן, שולח לכל התהליכים את הכל האוביקטים.  
כל תהליך מקצה זיכרון לכמות התמונות שצריך לקבל ואכן מקבל את כל התמונות לפי ה-chank המתאים שנשלח לו באמצעות “MPI\_Scatterv” ע"י תהליך 0.

אחרי שכל התהליכים קיבלו את כל הנתונים, עשיתי “Barrier” על מנת לסנכרן את כל התהליכים ושיתחילו יחד את החישוב כל אחד לפי תמונותיו.

**האלגוריתם המרכזי: (6 לולאות מקוננות)**לולאה ראשונה (חיצונית) רצה על מספר התמונות הספציפי שכל תהליך צריך לעבור, לולאה שנייה רצה על כל האובייקטים הקיימים, לולאה שלישית רצה על מספר השורות שמהם צריך להתחיל את הבדיקה בתמונה ( על מנת לא ליצור overflow ולחרוג מהתמונה בבדיקה), לולאה רביעית רצה על מספר העמודות שמהם צריך להתחיל את הבדיקה בתמונה, בתוך לולאה זאת מחשבים את סכום ערכי ה-matching (שזה פונקציה שבתוכה יש עוד 2 לולאות שרצות על השורות והעמודות של האובייקט הספציפי) ומחלקים במספר האיברים במטריצת האובייקט (החישוב המרכזי של הפרויקט כדי לבדוק אם האובייקט מתאים לתמונה), אם האובייקט מתאים לתמונה אז שומרים את מספר האוביקט והאינדקסים המתאימים שמהם מתחיל החישוב בתמונה, וכן את כמות האובייקטים שנמצאו בתמונה עד שמגיעים ל-3 אובייקטים (במידה ויש) ואז עוצרים את החיפוש.  
  
על לולאת התמונות (הלולאה החיצונית) שכל תהליך רץ בנפרד שמתי  
 “#pragma omp parallel for” כך שכל “thread” עובד על תמונה ספציפית במספר התמונות שאותו תהליך צריך לעבוד עליו, וכך מתייעלת העבודה משמעותית.

לאחר מכן כל התהליכים שולחים לתהליך 0 את התוצאות שהתקבלו בתמונות הספציפיות אותם הם בדקו. ותהליך 0 מקבל את כל התוצאות ומדפיס אותם לקובץ “output”, משחררים את כל הזיכרון שהוקצה ומסיימים את הפרויקט.