



קריית החינוך
פארק המדע
בית לערכים
למציאות ולחדשנות

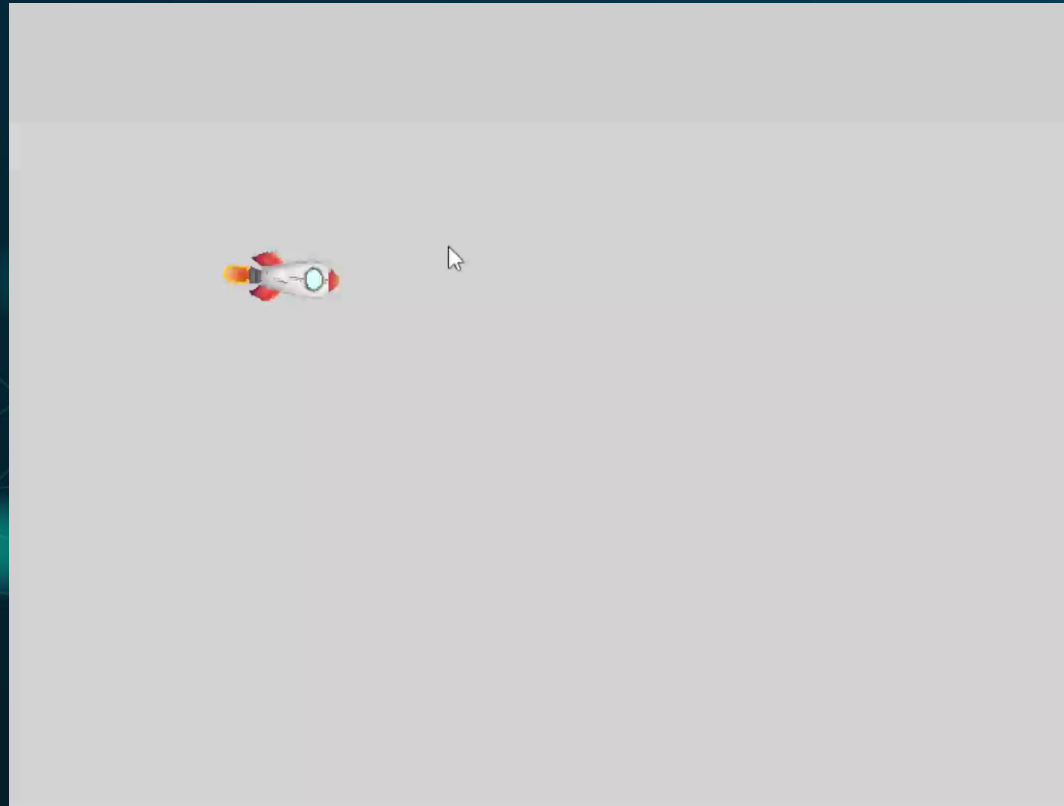
PyGame Vectors

גלעד מרקמן



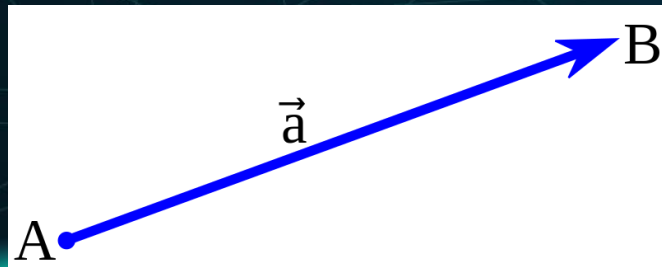
תנועה באמצעות וקטור

- נדגים כיצד נבצע תנועה של אובייקטים במרחב דו מימדי באמצעות וקטור מהירות.



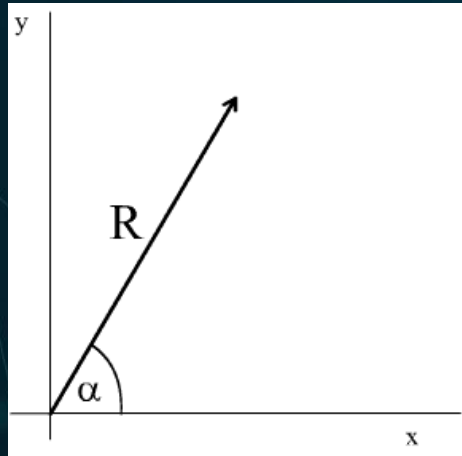
תנועה באמצעות וקטור

- וקטור הוא ערך בעל גודל וכיוון.
- לדוגמה: מהירות, תאוצה, כוח, שדה חשמלי ומעתק (דרך).
- לא די לנו לתאר מהירות של גוף מסויים באמצעות הגודל בלבד (20 קמ"ש). אנחנו צריכים גם לתאר את הכיוון של הגוף (צפון, דרום, למעלה, למטה וכד').
- אנחנו מייצגים וקטור במרחב באמצעות חץ, אשר אורכו מתאר את הגודל והכיוון בהתאם לכיוון החץ.



ייצוג ווקטור

- ייצג וקטור יכול להיעשות באמצעות שני ערכים: גודל וזווית.
- גודל – אורך החץ.
- זווית - נצייר מערכת צירים בבסיס הווקטור, ונמצא את הזווית עם ציר ה-x.

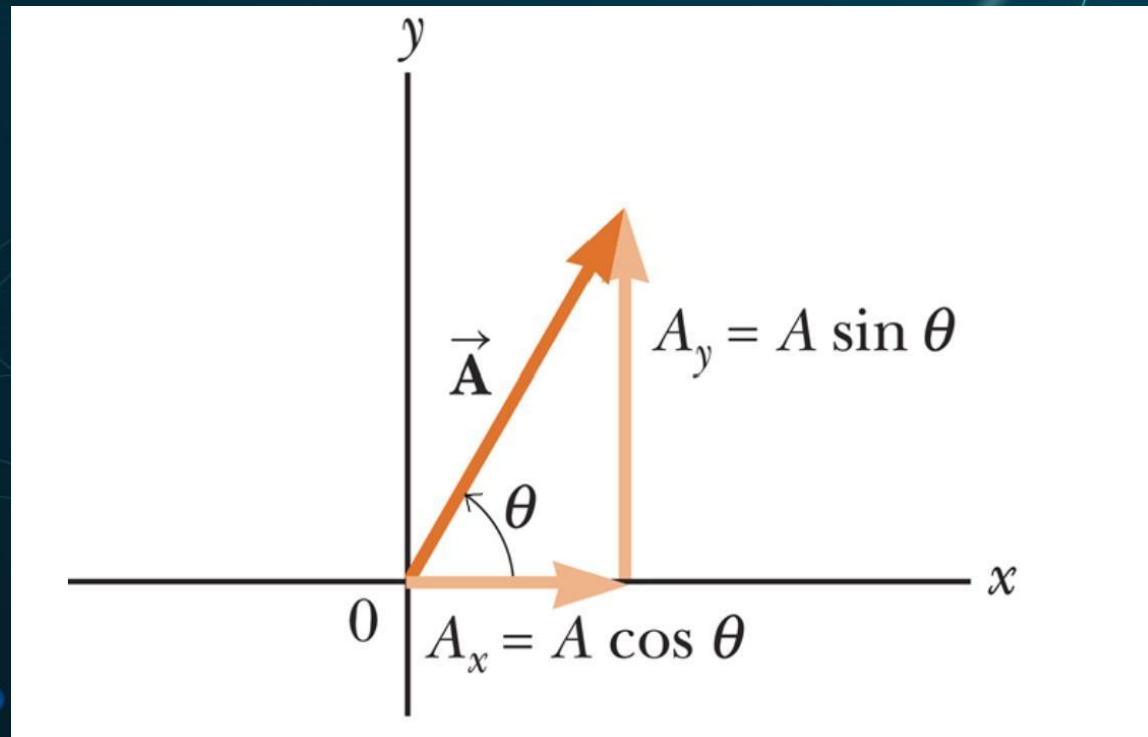


• בתמונה הווקטור ייוצג כ $v = (R, \alpha)$

- מיקום הווקטור אינו משנה את הווקטור. שני וקטורים זהים יכולים להיות מוצבים במקומות שונים במישור. תכונות הווקטור נקבעות לפי הגודל והכיוון, ולא מיקום הווקטור.

פירוק רכיבי הווקטור

- הווקטור A מתאר גוף הנע במישור במהירות $|A|$ ובכיוון זווית θ .
- ניתן לתאר תנועה זו בצורה שקולה לפי הצירים. הגוף נע על ציר x במהירות A_x ועל ציר y במהירות A_y .



חישוב רכיבי הוקטור

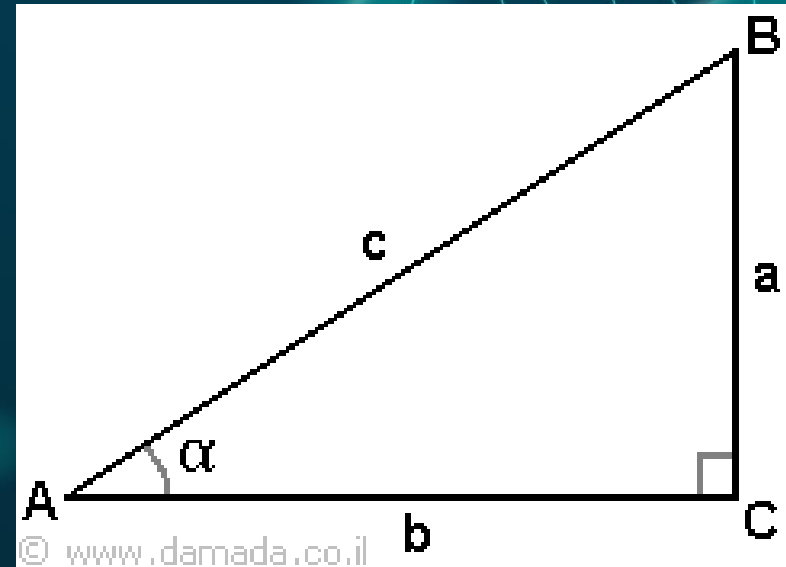
- חישוב רכיבי הוקטור נעשה באמצעות פונקציות טריגונומטריות, המוגדרות כך:

- $\sin(\alpha) = \frac{a}{c} \Rightarrow c * \sin(\alpha) = a$

- $\cos(\alpha) = \frac{b}{c} \Rightarrow c * \cos(\alpha) = b$

- $\sin(0^\circ) = 0; \sin(90^\circ) = 1$

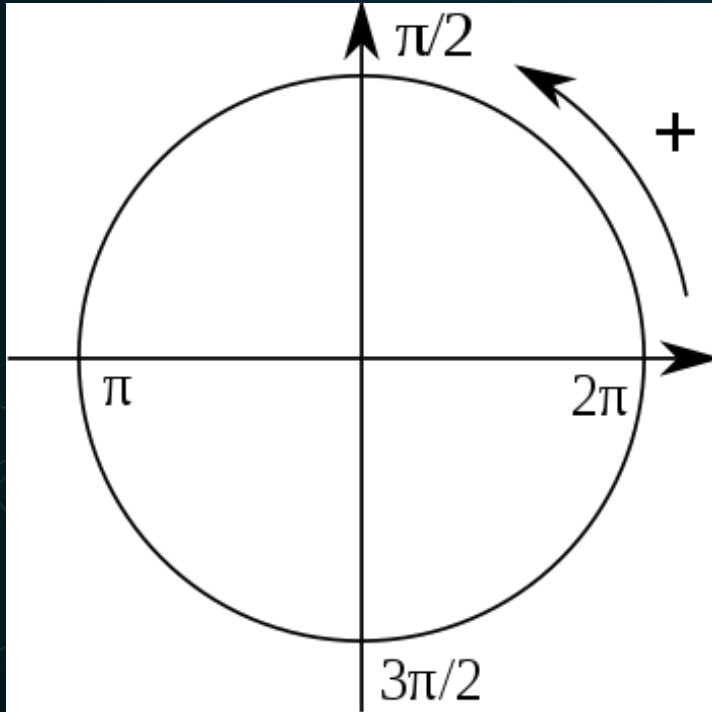
- $\cos(0^\circ) = 1; \cos(90^\circ) = 0$



תנועה באמצעות וקטור מהירות

- בהינתן לנו וקטור מהירות, הכולל את הגודל (מספר פיקסלים לפרים), נניע את התמונה על המסך באמצעות רכיבי הוקטור על ציר x – y .
- למשל אם נתבקש להניע גוף על המסך לפי וקטור מהירות של $10, 30^\circ = v$. הרי הוא ינוע כדלקמן:
 - $Speed_x = 10 * \cos(30^\circ) = 8.66$
 - $speed_y = 10 * \sin(30^\circ) = 5$
- בכל פריים תנוע התמונה 8.66 פיקסלים על ציר x ו 5 פיקסלים כל ציר y . בכך למעשה תנוע 10 פיקסלים בזווית של 30° .

ראדיאנים



• ניתן למדוד זוויות באמצעות שתי יחידות מידה:

• מעלות 90°

• ראדיאנים – המסומנים בחלק מהאות π – למשל $\frac{\pi}{2}$

• הראדיאנים מבוססים על הנוסחה של היקף המעגל $2\pi R$. על כן, $2\pi = 360^\circ$, $\pi = 180^\circ$.

• הפונקציות `math.sin`, `math.cos` בפייתון משתמשות בראדיאנים ולא במעלות. לכן יש לבצע המרה של הזווית.

• המרה: $radians = \frac{angle}{360} * 2\pi = \frac{angle}{180} * \pi$

• ניתן להמיר גם באמצעות הפונקציה `math.radians(angle)`.

הדגמה

פרוייקט ב GitHub

```
x, y = 200, 200  
speed, angle = 0, 0
```

```
space_ship = pygame.image.load("img/spacecraft.png")  
space_ship = pygame.transform.scale(space_ship, (100, 100))  
space_ship = pygame.transform.rotate(space_ship, -90)  
space_ship_rect = space_ship.get_rect()
```

```
run = True  
while (run):  
    for event in pygame.event.get():  
        if event.type == pygame.QUIT:  
            run = False  
        if event.type == pygame.KEYDOWN:  
            if event.key == pygame.K_RIGHT:  
                angle = angle - 5  
            if event.key == pygame.K_LEFT:  
                angle = angle + 5  
            if event.key == pygame.K_UP:  
                speed += 1  
            if event.key == pygame.K_DOWN:  
                speed -= 1  
            if event.key == pygame.K_SPACE:  
                speed = 0
```

```
screen.fill(LIGHTGRAY)
```

```
x += (math.cos(angle/360 * 2 * math.pi) * speed)  
x = x % WIDTH  
y -= (math.sin(math.radians(angle)) * speed)  
y = y % HEIGHT
```

```
space_ship_rotated = pygame.transform.rotate(space_ship, angle=angle)
```

```
space_ship_rect = space_ship.get_rect()  
space_ship_rect.center = x, y  
screen.blit(space_ship_rotated, space_ship_rect)
```

```
pygame.display.flip()  
clock.tick(FPS)
```