# Sprawozdanie
# Grafika komputerowa i komunikacja człowiek-komputer
# Laboratorium 4

Katsiaryna Kolyshko 276708

Dr. Inż. Jan Nikodem

# Wstęp teoretyczny

Zadanie laboratoryjne polega na dodawanie oświetlania na gotowy model jajka. Musieliśmy dodać nowe funkcje do naszego programu: Dodać dwa światła kolorowe, dając możliwość użytkowniku sterować nimi.

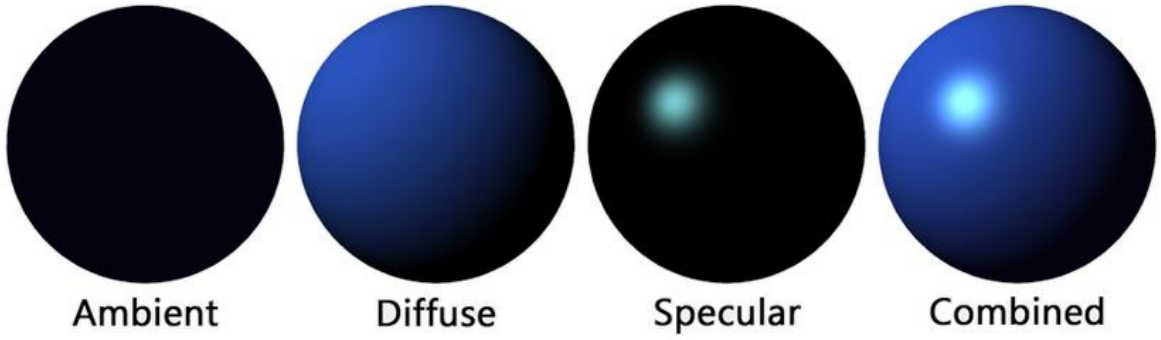
**Model Phonga**

Aby uzyskać oświetlenie w scenie 3D, należy określić, w jaki sposób obiekty reagują na światło. Istnieje wiele metod definiowania oświetlenia, jednak na potrzeby laboratorium wykorzystano *model Phonga*.

Model Phonga opisuje światło za pomocą trzech głównych składowych:

- *Światło otoczenia (ambient)* – reprezentuje światło otaczające o stałej intensywności. Nadaje obiektom kolor, który pochodzi od odbicia tej składowej.
- *Światło rozproszone (diffuse)* – opisuje światło emitowane ze źródła i padające na powierzchnię obiektów, co tworzy wrażenie ich oświetlenia.
- *Światło odbite (specular)* – symuluje biały połysk na powierzchniach odbijających, co pozwala łatwo określić, z którego miejsca pochodzi światło padające na obiekt.

Model ten jest często wykorzystywany do realistycznego renderowania, łącząc prostotę z wysoką jakością wizualną.




*Rysunek 1: Wizualizacja modelu Phonga*

Dla każdego obiektu odbijającego światło należy przeprowadzać osobne obliczenia. Proces ten opiera się na wektorach normalnych wyznaczanych dla każdego punktu opisującego powierzchnię obiektu. Takie podejście znacząco upraszcza skomplikowane obliczenia. Wektory normalne oblicza się zgodnie z poniższymi wzorami.

$$x_u = \frac{\partial x(u,v)}{\partial u} = \left(-450u^4 + 900u^3 - 810u^2 + 360u - 45\right) \cdot \cos(\pi v)$$

$$x_v = \frac{\partial x(u,v)}{\partial v} = \pi \cdot \left(90u^5 - 225u^4 + 270u^3 - 180u^2 + 45u\right) \cdot \sin(\pi v)$$

$$y_u = \frac{\partial y(u,v)}{\partial u} = 640u^3 - 960u^2 + 320u$$

$$y_v = \frac{\partial y(u,v)}{\partial v} = 0$$

$$z_u = \frac{\partial z(u,v)}{\partial u} = \left(-450u^4 + 900u^3 - 810u^2 + 360u - 45\right) \cdot \sin(\pi v)$$

$$z_v = \frac{\partial z(u,v)}{\partial v} = -\pi \cdot \left(90u^5 - 225u^4 + 270u^3 - 180u^2 + 45u\right) \cdot \cos(\pi v)$$

*Rysunek 2: Wzory opisujące wektory normalne*

Po obliczeniu wektoru normlanego, należy dokonać normalizacji, czyli podzielić wszystkie wartości przez długość wektora, co skutkuje otrzymaniem wektora o długości 1.

# Opis działania kodu

Nowe funkcje OpenGL wykorzystywane w programie:

• *glLightfv* — Używana do ustawiania właściwości źródła światła, takich jak kolor, intensywność i inne parametry. Dzięki tej funkcji, możemy dostosować właściwości oświetlenia, aby uzyskać realistyczne efekty świetlne w scenie 3D.

• *glLightf* — Służy do ustawiania pojedynczej właściwości źródła światła, takiej jak współczynniki tłumienia (attenuation). Jest to istotne dla regulowania, jak światło zmienia swoją intensywność w zależności od odległości od obiektu, co pozwala na tworzenie bardziej naturalnych efektów oświetlenia w scenach 3D.

• *glMaterialfv* — Umożliwia ustawienie właściwości materiałów, które określają, jak obiekt odbija światło. Dzięki tej funkcji można kontrolować takie parametry, jak kolor ambientalny, dyfuzyjny, spekulacyjny oraz połysk, co wpływa na wygląd obiektów w scenie i ich interakcję ze światłem.

Opis działania nowych funkcji:

```
58    # Light control variables
59    light_control_mode = 0  # 0 = control object, 1 = control lights
60    selected_light = 0      # 0 = GL_LIGHT0, 1 = GL_LIGHT1
```

Te dwie zmienne służą do sterowania światłami w scenie renderowanej. Zmienna `light_control_mode` określa, czy użytkownik steruje obiektem (`0`) czy światłami (`1`). Z kolei `selected_light` wskazuje, które światło jest aktualnie wybrane do modyfikacji: `0` oznacza `GL_LIGHT0`, a `1` oznacza `GL_LIGHT1`. Pozwala to na dynamiczne zmienianie ustawień świateł w scenie w zależności od trybu pracy.

```
68    # (u, v) coordinates for light positions in the matrix
69    light0_uv = [n // 3, n // 3]      # Position of the first light
70    light1_uv = [2 * n // 3, 2 * n // 3]  # Position of the second light
```

Te zmienne określają współrzędne (u, v) pozycji świateł na macierzy reprezentującej siatkę obiektu. light0_uv definiuje pozycję pierwszego światła jako jedną trzecią długości siatki zarówno w kierunku poziomym, jak i pionowym, podczas gdy light1_uv ustawia pozycję drugiego światła na dwóch trzecich tych długości. Dzięki temu światła są rozmieszczone proporcjonalnie względem wymiarów siatki.

```
169  def mouse_motion_callback(window, x_pos, y_pos):  1 usage  new *
170      global delta_x, delta_y, mouse_x_pos_old, mouse_y_pos_old
171      global light0_uv, light1_uv, light_control_mode, selected_light
172
173      delta_x = x_pos - mouse_x_pos_old
174      delta_y = y_pos - mouse_y_pos_old
175      mouse_x_pos_old = x_pos
176      mouse_y_pos_old = y_pos
177
178      # Sterowanie światłami myszką
179      if light_control_mode == 1:
180          if selected_light == 0:
181              light0_uv[0] = min(max(0, light0_uv[0] + int(delta_y / 5)), n)
182              light0_uv[1] = min(max(0, light0_uv[1] + int(delta_x / 5)), n)
183          else:
184              light1_uv[0] = min(max(0, light1_uv[0] + int(delta_y / 5)), n)
185              light1_uv[1] = min(max(0, light1_uv[1] + int(delta_x / 5)), n)
```

Funkcja *mouse_motion_callback* obsługuje ruch myszy, obliczając różnicę pozycji kursora w osi X (delta_x) i Y (delta_y) względem poprzedniej klatki, a następnie aktualizuje stare wartości pozycji kursora. Jeśli light_control_mode jest ustawiony na 1, ruch myszy steruje pozycją świateł na siatce. W zależności od wybranego światła (selected_light), zmieniane są współrzędne light0_uv lub light1_uv, gdzie przesunięcia są skalowane przez dzielenie przez 5, a wartości są ograniczone do przedziału od 0 do n, aby światła nie wychodziły poza zakres siatki.

```
212      if key == GLFW_KEY_T and action == GLFW_PRESS:
213          light_control_mode = 1 - light_control_mode  # Toggle light control mode
214
215      if key == GLFW_KEY_R and action == GLFW_PRESS:
216          selected_light = 1 - selected_light  # Toggle selected light source
```

Ten fragment kodu obsługuje zdarzenia klawiatury, pozwalając użytkownikowi przełączać tryb sterowania światłami oraz wybierać źródło światła. Jeśli użytkownik naciśnie klawisz T, zmienna light_control_mode zmienia się pomiędzy 0 (sterowanie obiektem) a 1 (sterowanie światłami). Naciśnięcie klawisza R przełącza zmienną selected_light pomiędzy 0 (pierwsze światło GL_LIGHT0) a 1 (drugie światło GL_LIGHT1), umożliwiając zmianę aktywnego źródła światła.

```
218      # Handle zooming
219      if key == GLFW_KEY_N and (action == GLFW_PRESS or action == GLFW_REPEAT):  # Zoom in
220          if e_button_state:  # Camera mode
221              if R > 1:
222                  R -= 0.05
223          else:  # Object mode
224              if scale < 3:
225                  scale += 0.05
226
227      if key == GLFW_KEY_M and (action == GLFW_PRESS or action == GLFW_REPEAT):  # Zoom out
228          if e_button_state:  # Camera mode
229              if R < 10:
230                  R += 0.05
231          else:  # Object mode
232              if scale > 0.3:
233                  scale -= 0.05
```

Ten fragment kodu obsługuje funkcję zoomowania, reagując na naciśnięcia klawiszy N i M. Klawisz N odpowiada za przybliżenie (zoom in), a M za oddalenie (zoom out). W trybie kamery (e_button_state aktywny) zmienna R kontrolująca odległość kamery od środka jest zwiększana lub zmniejszana w odpowiednich granicach (od 1 do 10). W trybie obiektu (e_button_state nieaktywny) zmienna scale odpowiadająca za skalowanie obiektu jest modyfikowana w granicach od 0.3 do 3, pozwalając na powiększanie lub zmniejszanie renderowanego obiektu.

```
422  def configure_light(light_id, position, diffuse, constant_attenuation=0.5, linear_attenuation=0.2):  4 usages  new *
423  #Funkcja pomocnicza do konfiguracji światła
424      glLightfv(light_id, GL_POSITION, position)
425      glLightfv(light_id, GL_DIFFUSE, diffuse)
426      glLightf(light_id, GL_CONSTANT_ATTENUATION, constant_attenuation)
427      glLightf(light_id, GL_LINEAR_ATTENUATION, linear_attenuation)
```

Funkcja configure_light służy do konfiguracji właściwości światła w scenie renderowanej za pomocą OpenGL, umożliwiając elastyczne ustawianie parametrów dla dowolnego źródła światła. Przyjmuje identyfikator światła (light_id), takie jak GL_LIGHT0 lub GL_LIGHT1, oraz jego pozycję (position), która definiuje, gdzie światło znajduje się w przestrzeni 3D. Oprócz tego można ustawić wartości światła rozproszonego (diffuse), które wpływają na kolor i intensywność światła odbitego od powierzchni obiektów w scenie.

Dodatkowe parametry obejmują tłumienie światła w zależności od odległości od źródła. Współczynnik tłumienia stałego (constant_attenuation) domyślnie wynosi 0.5 i określa podstawowe osłabienie światła niezależnie od odległości. Natomiast współczynnik tłumienia liniowego (linear_attenuation), domyślnie ustawiony na 0.2, powoduje stopniowy spadek intensywności światła wraz z odległością od źródła. Funkcja wykorzystuje funkcje OpenGL, takie jak glLightfv i glLightf, aby ustawić te parametry bezpośrednio w kontekście renderowania. Dzięki temu użytkownik może łatwo konfigurować różne źródła światła, minimalizując powtarzanie kodu.

```
440      # Włączanie oświetlenia
441      glEnable(GL_LIGHTING)
442      glEnable(GL_LIGHT0)
443      glEnable(GL_LIGHT1)
444      glEnable(GL_LIGHT2)
445
446      # Konfiguracja głównego źródła światła (GL_LIGHT0)
447      light0_position = [3.0, 3.0, 3.0, 1.0]  # Blisko obiektu
448      light0_diffuse = [0.75, 0.5, 1.0, 1.0]   # Lilac color
449      light0_ambient = [0.1, 0.0, 0.0, 1.0]
450      light0_specular = [1.0, 0.5, 0.5, 1.0]
451      glLightfv(GL_LIGHT0, GL_AMBIENT, light0_ambient)
452      glLightfv(GL_LIGHT0, GL_SPECULAR, light0_specular)
453      configure_light(GL_LIGHT0, light0_position, light0_diffuse)
454
455      # Konfiguracja drugiego źródła światła (GL_LIGHT1)
456      light1_position = [-3.0, 3.0, 3.0, 1.0]  # Blisko obiektu
457      light1_diffuse = [1.0, 1.0, 0.0, 1.0]  # Ywllo color
458      light1_ambient = [0.0, 0.1, 0.0, 1.0]
459      light1_specular = [0.5, 1.0, 0.5, 1.0]
460      glLightfv(GL_LIGHT1, GL_AMBIENT, light1_ambient)
461      glLightfv(GL_LIGHT1, GL_SPECULAR, light1_specular)
462      configure_light(GL_LIGHT1, light1_position, light1_diffuse)
```

Ten kod odpowiada za włączenie oświetlenia w scenie renderowanej w OpenGL oraz konfigurację dwóch źródeł światła. Na początku aktywowane są funkcje oświetlenia, włączając ogólne oświetlenie (GL_LIGHTING) oraz konkretne źródła światła (GL_LIGHT0, GL_LIGHT1, GL_LIGHT2). Dzięki temu światła będą miały wpływ na wygląd obiektów w scenie.

Dalej kod ustawia właściwości dwóch źródeł światła. Pierwsze źródło światła (GL_LIGHT0) jest ustawione w pozycji blisko obiektu. Dodatkowo, przypisane są wartości światła otoczenia (light0_ambient) i światła spektralnego (light0_specular). Te wartości wpływają na sposób, w jaki światło oddziałuje z powierzchniami obiektów, dając efekt rozpraszania i odbicia światła. Druga konfiguracja dotyczy GL_LIGHT1, które znajduje się w pozycji [-3.0, 3.0, 3.0, 1.0], również blisko obiektu, ale z żółtym światłem rozproszonym (light1_diffuse = [1.0, 1.0, 0.0, 1.0]). Podobnie jak w przypadku pierwszego światła, przypisane są wartości światła otoczenia (light1_ambient) oraz spektralne (light1_specular). W obydwu przypadkach dla każdego światła wywoływana jest funkcja configure_light, która ustawia pozycję, kolor światła rozproszonego oraz inne właściwości światła.

```
464     # Materiał obiektu
465     material_diffuse = [0.7, 0.4, 0.8, 1.0]
466     material_ambient = [0.6, 0.2, 0.3, 1.0]
467     material_specular = [1.0, 1.0, 1.0, 1.0]
468     material_shininess = [50.0]
469     glMaterialfv(GL_FRONT_AND_BACK, GL_DIFFUSE, material_diffuse)  # Używaj obu stron
470     glMaterialfv(GL_FRONT_AND_BACK, GL_AMBIENT, material_ambient)
471     glMaterialfv(GL_FRONT_AND_BACK, GL_SPECULAR, material_specular)
472     glMaterialfv(GL_FRONT_AND_BACK, GL_SHININESS, material_shininess)
```

Ten kod służy do ustawiania właściwości materiału obiektu w OpenGL, co wpływa na sposób, w jaki obiekt reaguje na światło. Właściwości materiału kontrolują sposób odbicia światła od obiektu, co ma wpływ na jego wygląd, w tym kolor, połysk oraz reakcję na światło otoczenia, rozproszone i odbite.

Wartość material_diffuse ustawiona na [0.7, 0.4, 0.8, 1.0] określa kolor obiektu w świetle rozproszonym. Obiekt będzie miał odcień fioletowo-różowy. Ta właściwość określa, jak obiekt odbija światło od źródła światła.

Wartość material_ambient ustawiona na [0.6, 0.2, 0.3, 1.0] definiuje kolor obiektu w świetle otoczenia. Światło otoczenia to stałe, niedokierunkowane światło, które jest obecne w całej scenie. Ta wartość wpływa na to, jak obiekt jest oświetlany w miejscach, które nie są bezpośrednio oświetlone przez źródło światła. Kolor jest przygaszony, różowawy, co wpływa na wygląd obiektu w cieniu lub w miejscach, gdzie nie pada bezpośrednie światło.

Wartość material_specular ustawiona na [1.0, 1.0, 1.0, 1.0] określa kolor odblasków spekularnych obiektu, czyli jasnych punktów, które pojawiają się, gdy światło odbija się od powierzchni obiektu pod określonym kątem. Wartość [1.0, 1.0, 1.0] oznacza czysty biały odblask, dzięki czemu odblaski będą białe, gdy obiekt odbije światło.

Wartość material_shininess ustawiona na [50.0] kontroluje, jak bardzo połyskliwy jest obiekt. Wyższe wartości połyskliwości powodują mniejsze i ostrzejsze odblaski, co sprawia, że obiekt wydaje się bardziej odbijający światło. Niższe wartości powodują większe i łagodniejsze odblaski, dając efekt matowej powierzchni. Wartość 50.0 oznacza stosunkowo błyszczącą powierzchnię.

*Dlaczego materiał jest potrzebny:*

Użycie właściwości materiału jest kluczowe dla tworzenia realistycznych obiektów w renderowaniu 3D. Definiując, jak obiekt reaguje na światło, możemy symulować rzeczywiste efekty, takie jak odbicia, połysk i sposób, w jaki obiekty zmieniają swój wygląd w zależności od oświetlenia. Bez właściwości materiału obiekt miałby domyślny, jednorodny wygląd, bez głębi i realizmu.

W tym kodzie funkcje glMaterialfv są używane do ustawiania właściwości materiału dla obu stron obiektu (GL_FRONT_AND_BACK), co oznacza, że właściwości materiału będą miały zastosowanie zarówno do przedniej, jak i tylnej strony obiektu. Jest to konieczne w przypadku renderowania obiektów

3D, gdzie obie strony mogą być widoczne z punktu widzenia kamery, zapewniając spójny wygląd obiektu niezależnie od tego, która strona jest zwrócona ku widzowi.

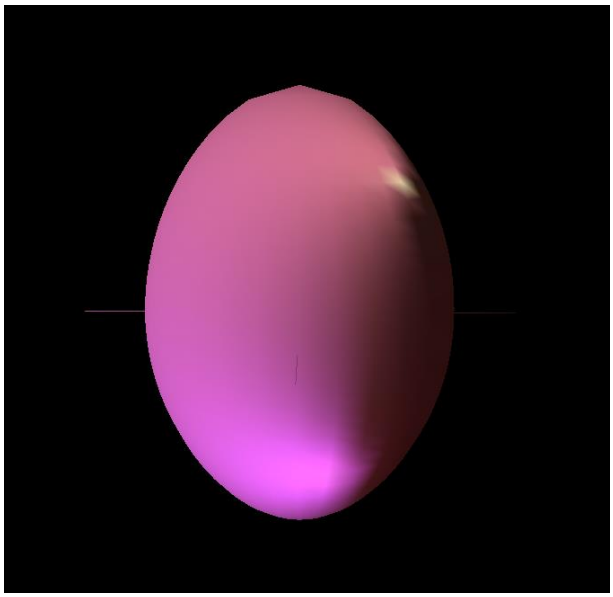
```
475    def set_dynamic_lights():  1 usage  new *
476        global matrix, light0_uv, light1_uv
477
478        # Wyznaczenie pozycji światła 0
479        light0_u, light0_v = light0_uv
480        light0_position = [
481            matrix[light0_u][light0_v][0],
482            matrix[light0_u][light0_v][1] - 5,  # Adjusting Y for visibility
483            matrix[light0_u][light0_v][2],
484            1.0,
485        ]
486
487        # Wyznaczenie pozycji światła 1
488        light1_u, light1_v = light1_uv
489        light1_position = [
490            matrix[light1_u][light1_v][0],
491            matrix[light1_u][light1_v][1] - 5,  # Adjusting Y for visibility
492            matrix[light1_u][light1_v][2],
493            1.0,
494        ]
```

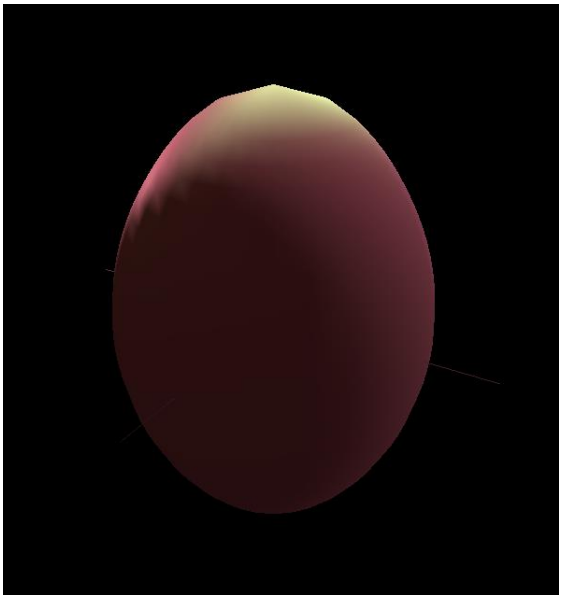
Funkcja set_dynamic_lights ma na celu obliczenie pozycji dwóch źródeł światła w przestrzeni 3D, korzystając z danych przechowywanych w macierzy. Pierwszym krokiem jest pobranie współrzędnych u i v dla obu świateł (light0_uv i light1_uv), które wskazują ich lokalizacje na macierzy. Następnie, dla każdego źródła światła, funkcja wyznacza jego pozycję, korzystając z wartości znajdujących się w macierzy w tych współrzędnych. Wartości te są wykorzystywane do ustalenia pozycji X i Z źródeł światła, natomiast pozycja Y jest modyfikowana o -5, aby zapewnić, że światło będzie lepiej widoczne w scenie. Ta modyfikacja w osi Y jest ważna, aby uniknąć sytuacji, w której światła mogłyby zostać umieszczone w obszarze, który nie jest dobrze widoczny w kontekście całej sceny 3D.

```
496        # Parametry światła 0 (lilac color)
497        light0_diffuse = [0.8, 0.6, 1.0, 1.0]  # Lilac
498        configure_light(GL_LIGHT0, light0_position, light0_diffuse)
499
500        # Parametry światła 1 (pastel lemon color)
501        light1_diffuse = [1.0, 1.0, 0.6, 1.0]  # Pastel Lemon
502        configure_light(GL_LIGHT1, light1_position, light1_diffuse)
503
```

W tym fragmencie kodu ustawiane są parametry dla dwóch źródeł światła. Dla światła 0 przypisany jest kolor fioletowo-liliowy, a dla światła 1 kolor pastelowego cytrynowego. Funkcja configure_light jest wywoływana dla obu źródeł światła, aby zastosować te kolory oraz pozycje, które zostały obliczone wcześniej, do odpowiednich źródeł światła OpenGL (GL_LIGHT0 i GL_LIGHT1).

Wyniki działanie programu:





```
Press Arrow Left or Arrow Right to switch between modes (Points, Lines, Triangles).
Use the mouse buttons to rotate and zoom the egg:
    Left Mouse Button: Rotate the camera/egg by holding the button down.
Press E to toggle between Camera mode and Egg mode.
Press G to increase the value of 'n' by 6 (maximum value is 120).
Press H to decrease the value of 'n' by 6 (minimum value is 6).
Press T to toggle light control mode (allows moving the selected light source with the mouse).
Press R to switch between the two light sources (active light will move with the mouse).
Press F to toggle face culling (enables or disables rendering of hidden surfaces).
Press N to zoom in (Camera mode) or increase the scale (Egg mode).
Press M to zoom out (Camera mode) or decrease the scale (Egg mode).
```