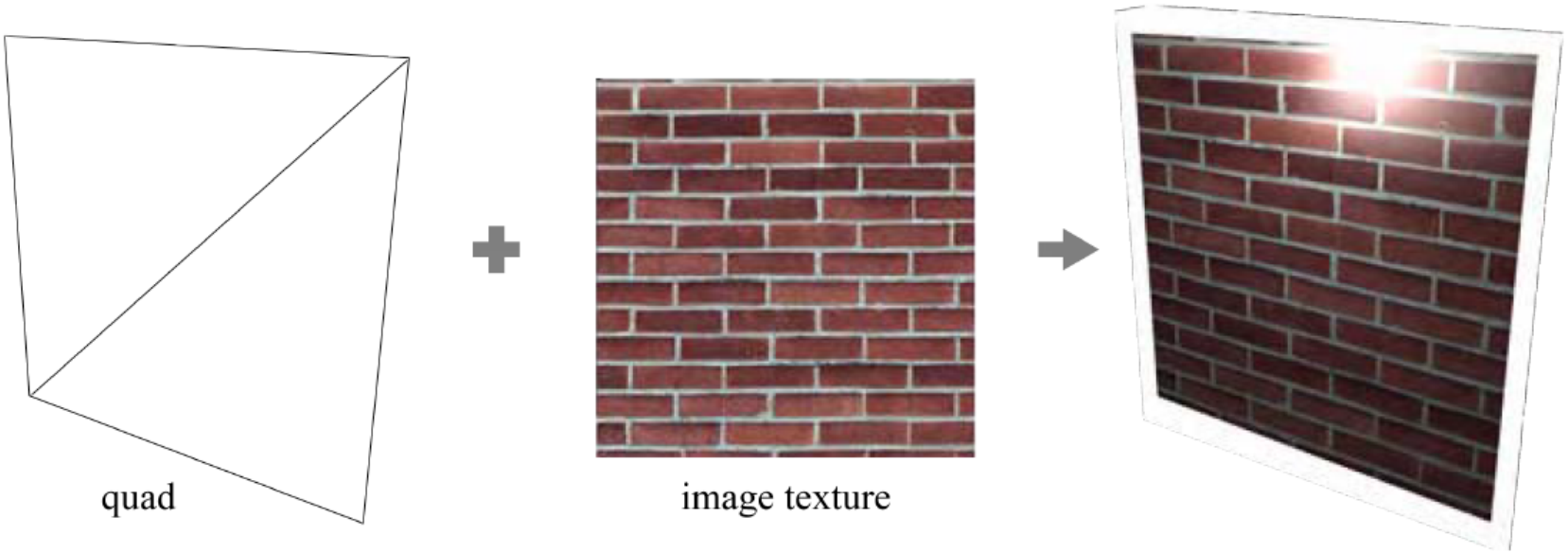
Grafika czasu rzeczywistego

Krzysztof Gdawiec

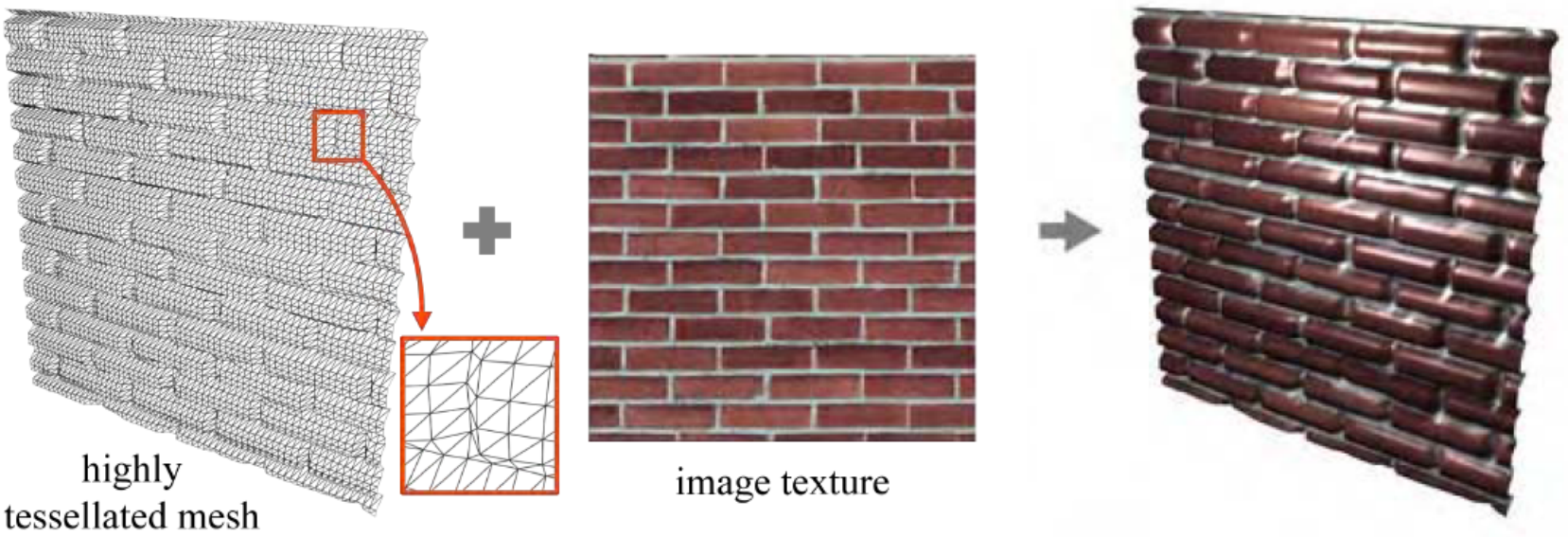
UNIWERSYTET ŚLĄSKI
INSTYTUT INFORMATYKI

Mapowanie nierówności

Używając teksturowania nie jesteśmy w stanie oddać nierówności powierzchni.



Jednym ze sposobów ukazania nierówności jest zwiększenie złożoności rysowanej geometrii.



Rozwiązaniem pośrednim, które obecnie jest stosowane, jest użycie prostej powierzchni (o małej rozdzielczości) i przechowywanie geometrycznych detali o wysokiej częstotliwości w teksturze. Technika ta nazywana jest mapowaniem nierówności (ang. bump mapping).

Pierwsze algorytmy mapowania nierówności zostały przedstawione przez Blinna w 1978 r. Zaobserwował on, że powierzchnia wydaje się mieć detale w małej skali jeśli podczas cieniowania zamienimy normala jego lekko zaburzoną wersją.

W mapowaniu nierówności zamiast używać tekstury do zmiany koloru w równaniu oświetlenia używamy tekstury do zmodyfikowania normala powierzchni.

Normal geometryczny powierzchni pozostaje bez zmian. Zmieniamy jedynie normala używanego w równaniu oświetlenia.

Ustawiając normala dla każdego wierzchołka sprawialiśmy, że powierzchnia wydawała się gładka. Teraz modyfikując normala dla każdego piksela zmieniamy sposób odbioru powierzchni wielokąta bez zmiany jego geometrii.

Zalety mapowania nierówności:

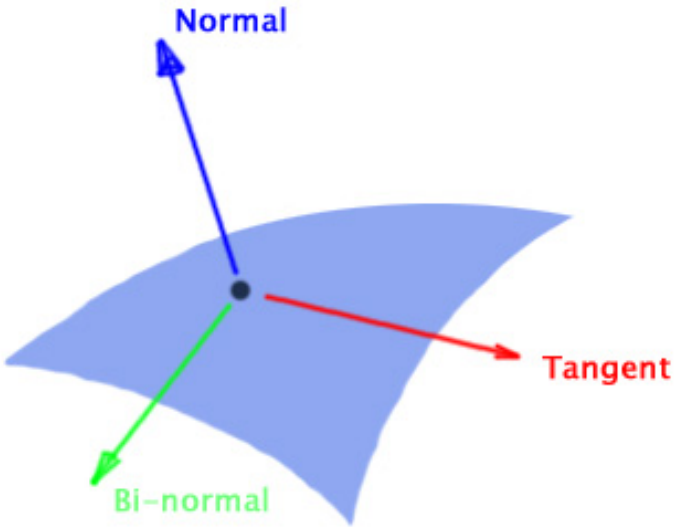
- zwiększenie wizualnej złożoności sceny bez zwiększania szczegółowości geometrii,
- uproszczenie sposobu tworzenia zawartości – szczegóły obiektu umieszcza się w teksturze zamiast w dokładniejszej siatce modelu,
- możliwość użycia różnych map nierówności dla różnych kopii tego samego obiektu umożliwia uzyskanie większej różnorodności, np. model budynku może być raz renderowany z mapą nierówności cegły a raz mapą nierówności tynku.

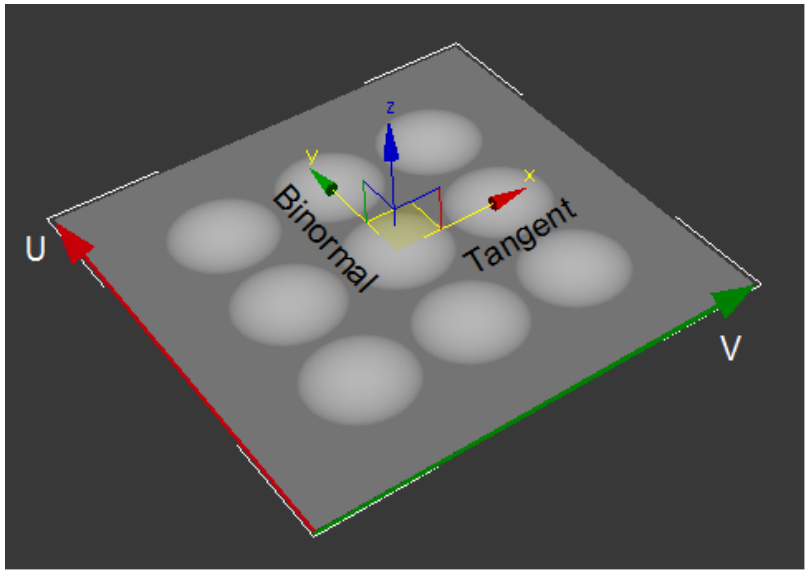
W mapowaniu nierówności normal musi zmieniać swój kierunek względem pewnego układu współrzędnych. Do tego celu najczęściej stosuje się tzw. układ przestrzeni stycznej.

Zaletą tego układu jest to, że tekstura, której używamy w mapowaniu nierówności może być użyta z dowolnym obiektem.

Układ przestrzeni stycznej jest używany do przetransformowania światła do przestrzeni położenia powierzchni (lub odwrotnie) w celu obliczenia efektu zmiany normala.

W przypadku siatek oprócz normala w wierzchołku przechowujemy wektor styczny (ang. tangent vector) i bistyczny (ang. bitangent vector). Wektor bistyczny często nazywany jest również wektorem binormalnym (ang. binormal vector).


Normal
Tangent
Bi-normal


z
y
x
Binormal
Tangent
U
V

Obliczanie bazy przestrzeni stycznej

Chcemy aby oś X (wektor styczny T) odpowiadała kierunkowi u, oś Y (wektor bistyczny B) odpowiadała kierunkowi v, a oś Z odpowiadała normalowi.

Załóżmy, że mamy trójkąt o wierzchołkach P_0, P_1, P_2, którym odpowiadają współrzędne tekstury (u_0, v_0), (u_1, v_1), (u_2, v_2). Szukamy wektorów T i B.

Ustalamy wierzchołek trójkąta P_0. Wówczas

$$
\begin{aligned}
Q_1 &= P_1 - P_0, \\
Q_2 &= P_2 - P_0, \\
(u_{10}, v_{10}) &= (u_1 - u_0, v_1 - v_0), \\
(u_{20}, v_{20}) &= (u_2 - u_0, v_2 - v_0).
\end{aligned}
$$

Musimy rozwiązać układ równań:

$$\begin{cases} Q_1 = u_{10}\,T + v_{10}\,B, \\ Q_2 = u_{20}\,T + v_{20}\,B. \end{cases}$$

Jest to układ sześciu równań liniowych z sześcioma niewiadomymi. Zapisując w formie macierzowej:

$$\begin{bmatrix} Q_{1x} & Q_{1y} & Q_{1z} \\ Q_{2x} & Q_{2y} & Q_{2z} \end{bmatrix} = \begin{bmatrix} u_{10} & v_{10} \\ u_{20} & v_{20} \end{bmatrix} \begin{bmatrix} T_x & T_y & T_z \\ B_x & B_y & B_z \end{bmatrix}$$

Rozwiązując układ otrzymujemy:

$$\begin{bmatrix} T_x & T_y & T_z \\ B_x & B_y & B_z \end{bmatrix} = \frac{1}{u_{10}v_{20} - u_{20}v_{10}} \begin{bmatrix} v_{20} & -v_{10} \\ -u_{20} & u_{10} \end{bmatrix} \begin{bmatrix} Q_{1x} & Q_{1y} & Q_{1z} \\ Q_{2x} & Q_{2y} & Q_{2z} \end{bmatrix}$$

W ten sposób otrzymujemy wektory (nieunormowane) T, B dla wierzchołka P_0 należącego do trójkąta P_0, P_1, P_2.

Aby obliczyć wektory T, B dla wierzchołka P_0 należącego do wielu trójkątów najpierw obliczamy wektory T, B dla każdego trójkąta, a następnie je dodajemy do siebie.

Kiedy mamy już wektory N, T, B możemy stworzyć macierz przejścia z układu przestrzeni stycznej do układu świata:

$$\begin{bmatrix} T_x & B_x & N_x \\ T_y & B_y & N_y \\ T_z & B_z & N_z \end{bmatrix}$$

Do przejścia w drugą stronę (od układu świata do układu przestrzeni stycznej) po prostu odwracamy tę macierz.

Wektory styczne otrzymane w przedstawiony sposób nie muszą być prostopadłe do siebie lub do normala. Wówczas macierz odwrotna nie jest równa transpozycji macierzy.

Używając ortogonalizacji Grama-Schmidta otrzymamy bazę ortogonalną:

$$T' = T - (N \cdot T)N,$$
$$B' = B - (N \cdot B)N - (T' \cdot B)T'.$$

Wektory T', B' nadal nie są unormowane.

Używając wektorów N, T', B' macierzą odwrotną do macierzy przejścia z układu przestrzeni stycznej do układu świata będzie jej macierz transponowana.

Nie musimy przechowywać dla każdego wierzchołka bistycznej ponieważ iloczyn wektorowy $N \times T'$ może być użyty do otrzymania mB', gdzie $m = \pm 1$ reprezentuje „skrętność" układu stycznego.

„Skrętność" musimy przechowywać dla każdego wierzchołka. Wartość m jest równa wyznacznikowi macierzy:

$$\begin{bmatrix} T'_x & T'_y & T'_z \\ B'_x & B'_y & B'_z \\ N_x & N_y & N_z \end{bmatrix}.$$

Wygodnie jest przechowywać wartość m jako czwartą współrzędną wektora T'. Wówczas bistyczna obliczana jest następująco:

$$B' = T'_w(N \times T'),$$

gdzie w iloczynie wektorowym pomijamy współrzędną w.

Mając macierz przejścia od układu świata do układu stycznego używamy jej do przeliczenia (dla każdego wierzchołka) wektora światła i wektora widoku używanych przy obliczeniach związanych z oświetleniem.

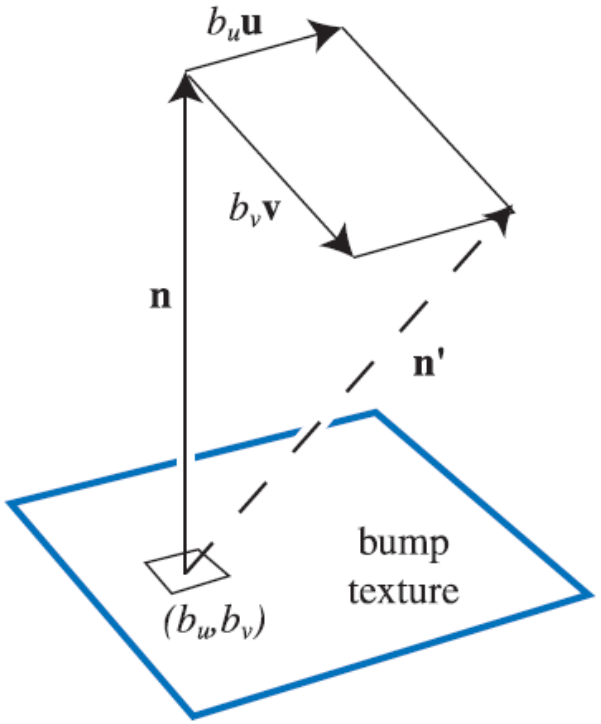
Mając macierz przejścia od układu świata do układu stycznego używamy jej do przeliczenia (dla każdego wierzchołka) wektora światła i wektora widoku używanych przy obliczeniach związanych z oświetleniem.

Algorytm Blinna

W oryginalnej metodzie Blinna przechowujemy dla każdego punktu dwie wartości ze znakiem b_u, b_v. Odpowiadają one wielkości o jaką mamy zmienić normala wzdłuż kierunków osi obrazu u, v.

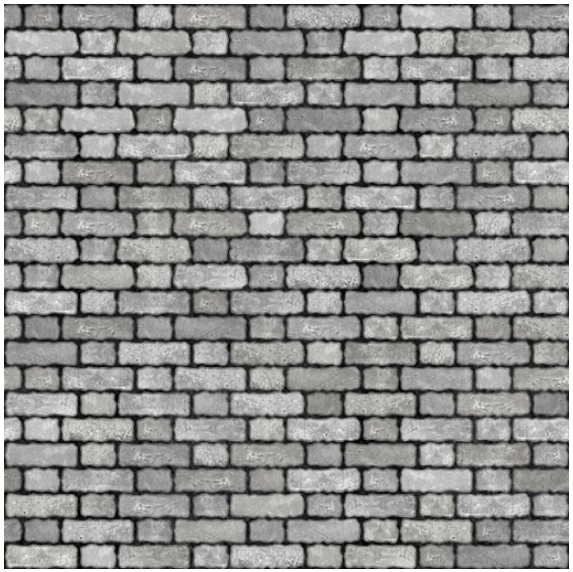
Wektory $b_u u$, $b_v v$ są dodawane do normala.

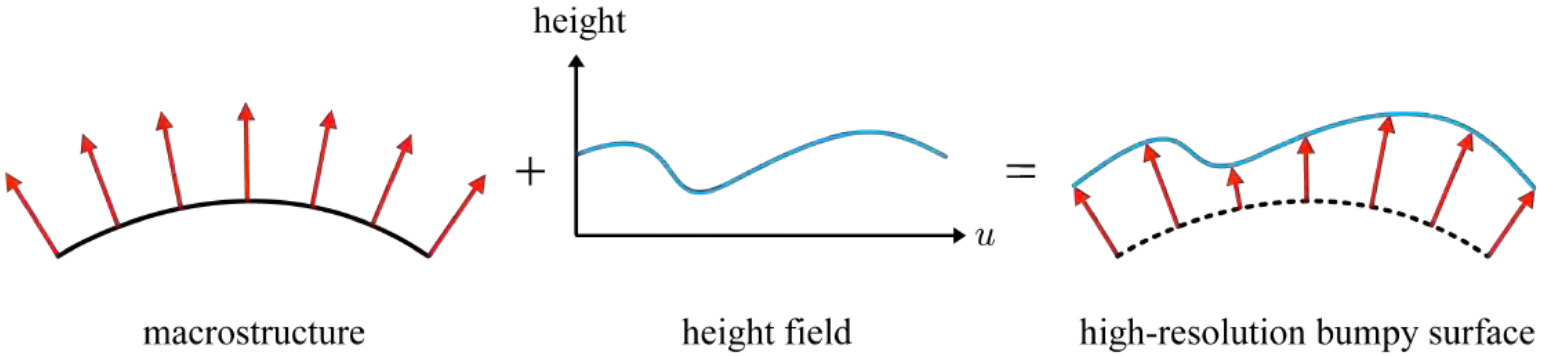
Ten typ mapowania nierówności nazywamy mapowaniem nierówności wektora przesunięcia (ang. offset vector bump mapping).


$b_u\mathbf{u}$
$b_v\mathbf{v}$
$\mathbf{n}$
$\mathbf{n}'$
bump
texture
(b_u, b_v)

Drugim algorytmem przedstawionym przez Blinna jest algorytm z użyciem pola wysokości.

Najczęściej pole wysokości przechowywane jest w postaci obrazu w odcieniach szarości. Kolor biały odpowiada najwyższemu punktowi, a kolor czarny najniższemu.





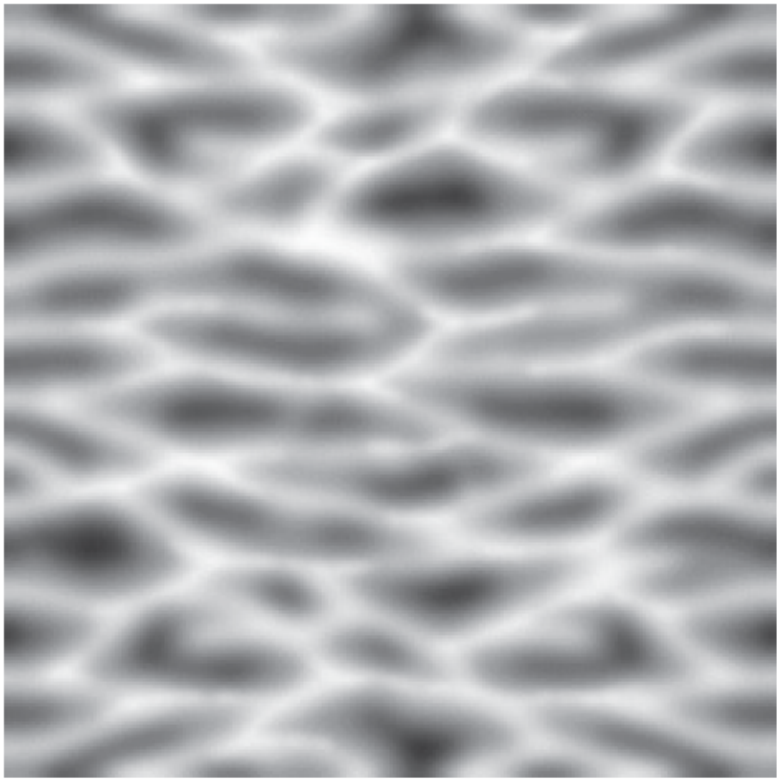
Niech $h(i,j)$ oznacza wysokość we współrzędnych (i,j).
Wyznaczamy wektory U, V następująco:

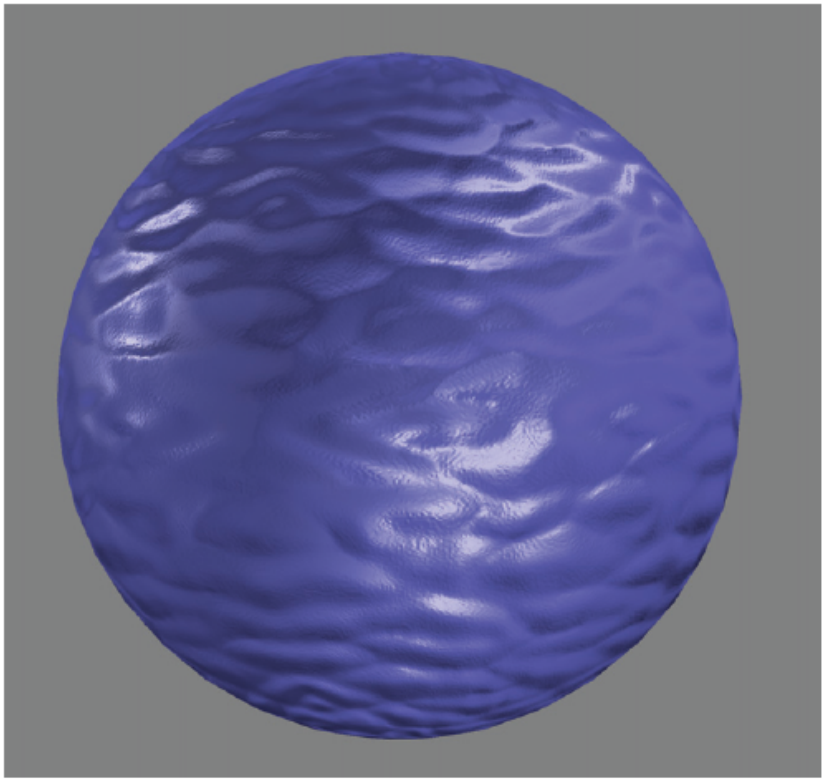
$$U(i,j) = (1, 0, ah(i+1,j) - ah(i-1,j)),$$
$$V(i,j) = (0, 1, ah(i,j+1) - ah(i,j-1)),$$

gdzie a jest to współczynnik skali służący do zmiany wysokości.
Wówczas normal wynosi:

$$N(i,j) = \frac{U(i,j) \times V(i,j)}{\|U(i,j) \times V(i,j)\|} = \frac{(-U(i,j)_z, -V(i,j)_z, 1)}{\sqrt{U(i,j)_z^2 + V(i,j)_z^2 + 1}}$$

Mapa wysokości i obiekt wyrenderowany z użyciem tej mapy do mapowania nierówności.





Mapowanie normali (ang. normal mapping)

W technice tej w teksturze przechowujemy bezpośrednio normala. Jest to preferowana technika dla obecnych kart graficznych ponieważ koszt przechowywania trzech składowych dla normala kontra dwóch składowych przemieszczenia czy jedenj wartości wysokości nie stanowi problemu. Redukuje to również ilość obliczeń, które musimy wykonać w shaderze fragmentów.

Mapa normali koduje współrzędne (x, y, z) do przedziału $[-1, 1]$, np. dla 8-bitowej tekstury współrzędna x o wartości 0 reprezentuje -1.0 a 255 reprezentuje 1.0.

Oryginalnie technika ta przedstawiona została używając mapy normali w przestrzeni świata. W takim przypadku dla każdego fragmentu pobieramy normala z mapy normali i używamy go do obliczeń związanych z oświetleniem. Podejście to ma tę wadę, że po zmianie orientacji obiektu, deformacji trzeba stworzyć nową mapę normali.

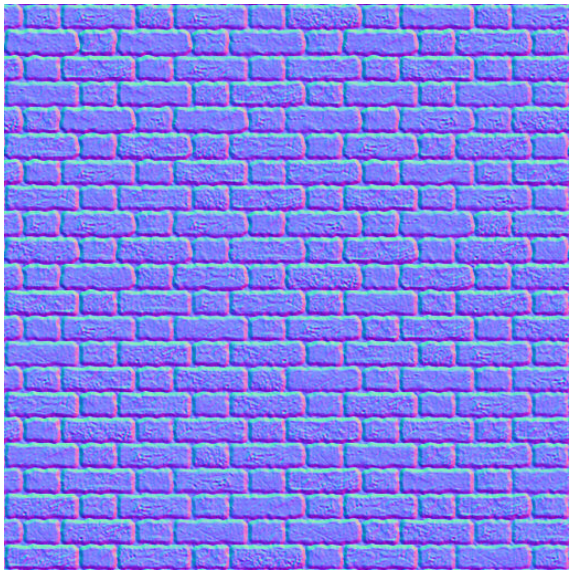
Oryginalnie technika ta przedstawiona została używając mapy normali w przestrzeni świata. W takim przypadku dla każdego fragmentu pobieramy normala z mapy normali i używamy go do obliczeń związanych z oświetleniem. Podejście to ma tę wadę, że po zmianie orientacji obiektu, deformacji trzeba stworzyć nową mapę normali.

Mapa normali może również być zdefiniowana w przestrzeni obiektu. Wówczas taka mapa jest dobra jeśli obiekt poddajemy przekształceniom sztywnym, ale nie nadaje się jeśli obiekt zostanie zdeformowany za pomocą dowolnej deformacji. Ponadto takie mapy normali nie mogą być współdzielone przez różne obiekty. Obliczenia oświetlenia muszą być wykonane w przestrzeni obiektu.

Najczęściej zaburzony normal jest podawany w przestrzeni stycznej, tzn. relatywnie do powierzchni.

Pozwala to na deformację powierzchni oraz współdzielenie tej samej mapy normali pomiędzy różnymi obiektami.

Wadą map normali w przestrzeni stycznej jest to, że musimy wykonać więcej obliczeń w shaderze ze względu na to, że układ odniesienia zmienia się na powierzchni.



Normale są przechowywane jako tekstury RGB, gdzie poszczególne składowe odpowiadają współrzędnym x, y, z. Komponent z jest zazwyczaj równy 1.0 lub bliski tej wartości. Z tego powodu kolorem dominującym w mapie normali jest kolor niebieski.

Przy obliczaniu oświetlenia używając standardowego modelu zarówno powierzchnia jak i światło muszą znajdować się w tym samym układzie współrzędnych: stycznym, obiektu lub świata.

Jedną z metod jest przetransformowanie wszystkich kierunków światła (patrząc z wierzchołka) do przestrzeni stycznej i interpolacja tych przetransformowanych wektorów dla trójkąta. Tak powstałe wektory wraz z normalem odczytanym z mapy normali używane są następnie do obliczenia oświetlenia.

Takie podejście działa ponieważ przy obliczeniach interesuje nas relatywny kierunek światła z punktu dla którego robimy obliczenia, a nie ich absolutne położenie w przestrzeni. Zatem kierunek światła zmienia się w małym stopniu i dlatego możemy go interpolować dla trójkąta.

Dla pojedynczego światła takie podejście jest również mniej kosztowne niż transformacja normala z mapy normali do przestrzeni świata dla każdego piksela.

Jeśli używamy więcej niż kilku świateł bardziej efektywne jest transformowanie normala do przestrzeni świata. W tym przypadku zamiast interpolować dużą liczbę kierunków światła dla trójkąta wykonujemy tylko jedną transformację normala. Ponadto jeśli dodajemy odbicia, to normal musi być w przestrzeni świata żeby obliczyć kierunek odbicia.

Mapowanie paralaksy (ang. parallax mapping)

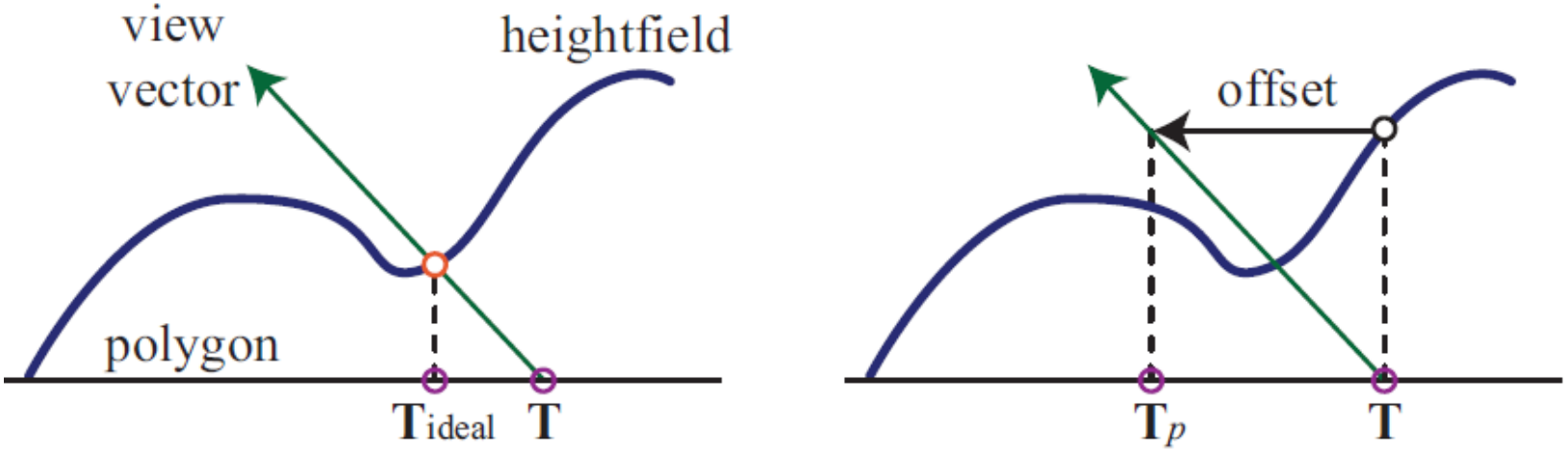
Problemem w mapowaniu normali jest to, że nierówności nigdy nie blokują się nawzajem. Np. jeśli patrzymy na ceglaną ścianę, to pod pewnym kątem nie zobaczymy zaprawy pomiędzy cegłami.

Z tego powodu w 2001 Kaneko wprowadził technikę mapowania paralaksy. Paralaksa odnosi się do idei, że pozycje obiektów zmieniają się relatywnie do siebie wraz ze zmianą położenia obserwatora. Wraz z ruchem obserwatora nierówności powinny zasłaniać się nawzajem, tzn. wydawać się, że mają wysokość.

Kluczowym pomysłem w mapowaniu paralaksy jest aby na podstawie zbadania wysokości tego co zostało uznane za widoczne „zgadnąć" co powinno być widoczne.

W mapowaniu paralaksy nierówności są przechowywane w mapie wysokości.

Kiedy patrzymy na powierzchnię w danym pikselu pobieramy odpowiednią wysokość z mapy wysokości i używamy jej aby przesunąć współrzędne tekstury w celu otrzymania innej części powierzchni i pobrania np. wektora normalnego z mapy normali. Wartość przesunięcia obliczana jest na podstawie odczytanej wysokości i kąta kierunku patrzenia na dany punkt.



Wartości wysokości mogą być przechowywane w osobnej teksturze albo jako kanał alfa jakiejś innej tekstury.

Wartości wysokości należy przeskalować i przesunąć przed użyciem do przesunięcia wartości. Skala określa jak wysokie jest nasze pole wysokości ponad lub pod powierzchnią, zaś przesunięcie daje nam „poziom morza”, na którym nie ma żadnego przesunięcia.

Wartości wysokości mogą być przechowywane w osobnej teksturze albo jako kanał alfa jakiejś innej tekstury.

Wartości wysokości należy przeskalować i przesunąć przed użyciem do przesunięcia wartości. Skala określa jak wysokie jest nasze pole wysokości ponad lub pod powierzchnią, zaś przesunięcie daje nam „poziom morza", na którym nie ma żadnego przesunięcia.

Mając pozycję p, odczytaną wysokość h oraz znormalizowany wektor widoku v (o wysokości v_z i składowych poziomych v_{xy}) pozycję po przesunięciu p_{adj} otrzymujemy następująco:

$$p_{adj} = p + \frac{h \cdot v_{xy}}{v_z}.$$

Wektor widoku musi być wyrażony w przestrzeni stycznej.

Wartości p_{adj} następnie używamy do odczytania np. normala z mapy normali, koloru itp.

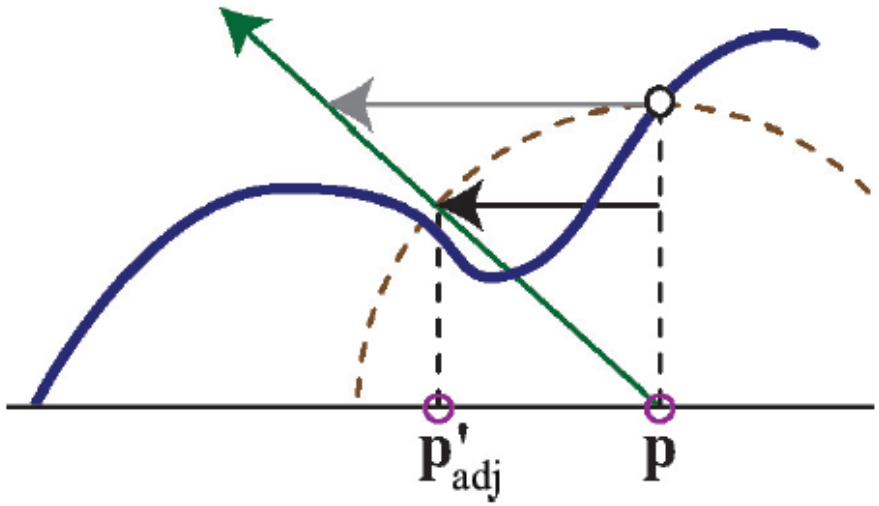
Wartości p_{adj} następnie używamy do odczytania np. normala z mapy normali, koloru itp.

Taka aproksymacja przesunięcia działa dość dobrze jeśli wysokości nierówności zmieniają się relatywnie wolno.

Aproksymacja ta nie sprawdza się przy „płytkich” kątach patrzenia, tzn. kiedy wektor widoku jest bliski horyzontowi powierzchni. Wówczas mała zmiana wysokości powoduje duże przesunięcie współrzędnych tekstury.

Aby zmniejszyć wpływ tego problemu Welsh w 2004 wprowadził pojęcie ograniczenia przesunięcia. Pomysł polega na ograniczeniu wartości przesunięcia tak aby nigdy nie była większa niż odczytana wysokość. Wówczas wzór na przesunięte współrzędne przyjmuje postać:

$$p'_{adj} = p + h \cdot v_{xy}.$$



Przy „stromych” kątach patrzenia równanie to jest prawie identyczne do poprzedniego, bo v_z jest bliskie 1.

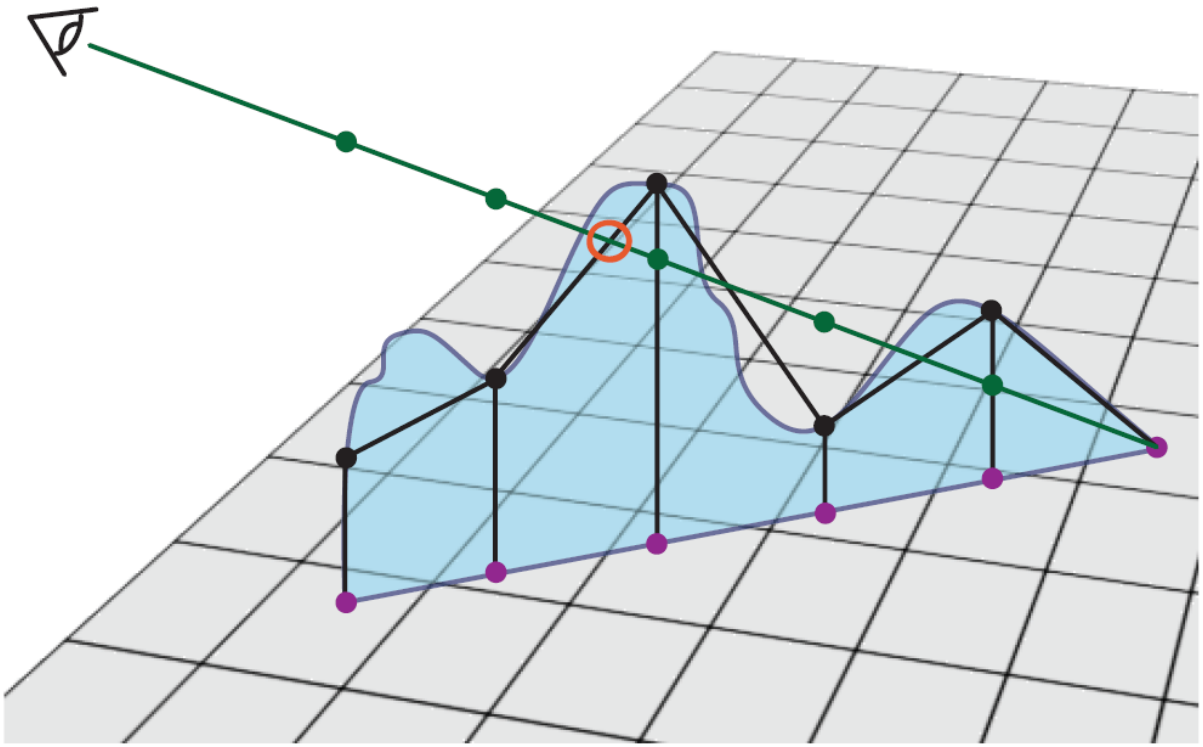
Pomimo swych wad mapowanie paralaksy z ograniczeniem przesunięcia kosztuje kilka dodatkowych instrukcji w shaderze fragmentów, ale daje lepszej jakości obrazy w porównaniu do mapowania normali. Z tego względu technika ta znalazła szerokie zastosowanie we współczesnych grach.

Mapowanie płaskorzeźby (ang. relief mapping)

Mapowanie paralaksy daje nam pewną aproksymację efektu pola wysokości, ale chcielibyśmy dostać to co jest widoczne w danym pikselu, tzn. to co jest na pierwszym przecięciu wektora widoku i pola wysokości.

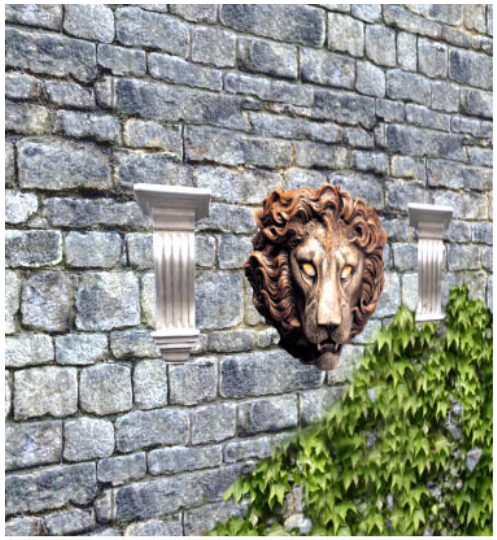
Mniej więcej w tym samym czasie wprowadzono podobne metody realizujące to zadanie. Nazwano je: parallax occlusion mapping (Brawley, Z., Tatarchuk, N., 2004), relief mapping (Policarpo, F., 2004), steep parallax mapping (McGuire, M., McGuire, M., 2005).

Pomysł polega na testowaniu ustalonej liczby próbek tekstury branych wzdłuż zrzutowanego wektora. Przy małym kącie padania najczęściej bierzemy więcej próbek, aby nie przegapić punktu przecięcia.



Dla każdej lokalizacji na teksturze pobieramy wartość i sprawdzamy czy jesteśmy powyżej czy poniżej pola wysokości.

Kiedy znajdziemy próbkę, która znajduje się poniżej pola wysokości używamy jej oraz próbki poprzedniej (znajduje się powyżej pola wysokości) do znalezienia miejsca przecięcia. Miejsce to później jest używane w dalszych obliczeniach związanych z cieniowaniem powierzchni.



Texture Mapped

Normal Mapped

Parallax Mapped

Steep Parallax Mapped