Laboratorium 12: „Podstawy grafiki 2D - biblioteka Allegro”

mgr inż. Leszek Ciopiński
dr inż. Arkadiusz Chrobot

4 września 2026

1. Wprowadzenie

Współcześnie, większość urządzeń elektronicznych zdolnych jest do wyświetlania zaawansowanej grafiki. Aby ułatwić programiście jej przygotowywanie dostępnych jest wiele bibliotek, różniących się między sobą możliwościami i przeznaczeniem. Jedną z takich bibliotek jest `Allegro`, która jest przeznaczona do projektowania gier 2D.

2. Instalacja biblioteki

Instalacja biblioteki pokazana będzie na przykładzie środowiska Code::Blocks w systemie MS Windows.

2.1. Instalacja plików

Ze strony `https://www.allegro.cc/files/?v=4.4` należy pobrać plik `allegro-4.4.2-mingw-4.4.0.zip` i zdekompresować go. Następnie zawartość folderu `bin` należy skopiować pod adres `C:\Program Files (x86)\CodeBlocks\MinGW\bin`, folderu `include` pod adres: `C:\Program Files (x86)\CodeBlocks\MinGW\include`, a folderu `lib` pod adres `C:\Program Files (x86)\CodeBlocks\MinGW\lib`

2.2. Konfiguracja projektu w środowisku Code::Blocks

W środowisku Code::Blocks należy przekazać informację o używanej dodatkowo bibliotece. Z tego powodu konieczne jest pracowanie z wykorzystaniem projektów – nie da się kompilować pliku z jego pominięciem. Po utworzeniu projektu, należy zaznaczyć jego nazwę i otworzyć menu kontekstowe (najczęściej prawym przyciskiem myszy). Należy wybrać polecenie `Build options...`. W nowo otwartym oknie `Project build options` należy przejść do karty `Linker settings`. W polu `Link libraries` należy dodać adres biblioteki `C:\Program Files (x86)\CodeBlocks\MinGW\lib\liballeg.a`. Na koniec potwierdzamy wybór przyciskiem `OK`.

2.3. Hello World!

Po wykonaniu czynności opisanych powyżej, możliwe jest skompilowanie projektu. Na Listingu 1 zaprezentowano najprostszą aplikację wykorzystującą bibliotekę. Czarny ekran oznacza w tym przypadku prawidłowy wynik działania programu.

```
#include <allegro.h>
int main(void)
{
    allegro_init();
    install_keyboard();
    set_gfx_mode( GFX_AUTODETECT, 640, 480, 0, 0);
    readkey();
    return 0;
}
END_OF_MAIN();
```

Listing 1: Program HelloWorld z wykorzystaniem biblioteki `allegro`.

3. Funkcje

Rozpoczynając tryb pracy z biblioteką, można też skonfigurować kilka dodatkowych parametrów pracy.

3.1. Funkcje inicjujące

int allegro_init(); Makro inicjujące bibliotekę `Allegro`.

int install_keyboard(); Uruchamia obsługę klawiatury przez bibliotekę `Allegro`. Należy zauważyć, że od momentu inicjacji biblioteki nie można już używać standardowych funkcji systemowych ani funkcji języka `C` do odczytu danych z klawiatury. Ze względu na różnice implementacyjne na różnych systemach, nie można mieć 100% pewności, że odczyt danych z klawiatury będzie możliwy, dopóki nie zostanie uruchomione środowisko graficzne biblioteki funkcją `set_gfx_mode()`.

void set_color_depth(int depth); Określa głębokość koloru, jaki będzie używany przez funkcję biblioteki. Dopuszczalnymi wartościami są: `8` (wartość domyślna), `15`, `16`, `24`, `32`.

int set_gfx_mode(int card, int w, int h, int v_w, int v_h); Uruchamia graficzne środowisko biblioteki. Pierwszym parametrem jest informacja o platformie graficznej. Aby pozwolić bibliotece automatycznie wybrać właściwy sterownik, należy przekazać tu wartość `GFX_AUTODETECT`. Następne dwa parametry określają szerokość i wysokość tworzonego okna w pikselach. Ostatnie dwa parametry określają rozmiar "wirtualnego okna", które jest używane do zamiany przy odświeżaniu ekranu. Nie każdy rozmiar jest akceptowany przez sterowniki. Aby uzyskać dopasowanie do każdego okna, należy przekazać tu wartości `0` i `0`.

void allegro_message(const char *text_format, ...); Wyświetla komunikat w formie podanego ciągu. Funkcja ta jest zależna od środowiska, dlatego nie zawsze może działać. Nie wolno jej jednak używać, jeżeli program pracuje już w trybie graficznym.

```
void  inicjuj(void){
^^Iif(!allegro_init()){
^^I^^Iprintf("Błąd inicjacji bibliotek i Allegro !\n");
^^I^^Ireturn;
^^I}
^^Iif(! install_keyboard()){
^^I^^Iallegro_message("Błąd inicjacji bibliotek i Allegro !\n");
^^I^^Ireturn;
^^I}
^^Iset_color_depth(32);
^^Iif(!set gfx mode (AL GFX AUTODETECT FULLSCREEN, 1366, 768, 0, 0 ){
^^I^^Iallegro_message("Błąd ustawiania trybu gra fizycznego !\n") ;
^^I^^Ireturn;
^^I}
}
```

Listing 2: Program prezentujący konfigurowanie biblioteki `Allegro`.

3.2. Tworzenie obrazu

Współrzędne ekranu i dowolnej bitmapy liczone są względem lewego górnego rogu. Piksel znajdujący się w tym miejscu ma współrzędne o wartości (0, 0).

typedef struct BITMAP Struktura danych wykorzystywana jako bufor ramki.

extern BITMAP *screen; Wskaźnik globalny do głównego obrazu, który jest aktualnie wyświetlany na ekranie.

void putpixel(BITMAP *bmp, int x, int y, int color); Ustawia kolor piksela na pozycji `x`, `y` na kolor określony w trzecim parametrze, który jest zgodny z aktualnie przyjętym trybem rysowania. Kolor można uzyskać przy pomocy funkcji `makecol()`. Istnieją też makra: `_putpixel()`, `_putpixel15()`, `_putpixel16()`, `_putpixel24()`, `_putpixel32()`, które są zaimplementowane jako polecenia assemblerowe. W związku z tym działają dużo szybciej. Nie sprawdzają one jednak poprawności ich użycia, dlatego nieodpowiednie ich wywołanie może doprowadzić do awarii programu.

int makecol(int r, int g, int b); Przekształca kolor podany w formacie niezależnym od sprzętu na aktualnie wykorzystywany format graficzny. Wszystkie wartości powinny być w zakresie od `0` do `255`.Odpowiednio zakodowany kolor zwracany jest jako wynik z funkcji.

void line(BITMAP *bmp, int x1, int y1, int x2, int y2, int color); Rysuje linię w kolorze `color` od punktu `(x1, y1)` do punktu `(x2, y2)`.

void circle(BITMAP *bmp, int x, int y, int radius, int color); Rysuje okrąg o określonym punkcie środka i promieniu.

void rect(BITMAP *bmp, int x1, int y1, int x2, int y2, int color); Rysuje prostokąt w kolorze `color` od punktu `(x1, y1)` do punktu `(x2, y2)`.

3.3. Obsługa animacji

Grafika 2d może przedstawiać obraz statyczny lub dynamiczny. Aby animacje były płynne obraz musi być wyświetlany z częstotliwością co najmniej 24 klatek na sekundę. Bibliotek `Allegro` wspiera tworzenie animacji poprzez dostarczenie mechanizmu obsługi tak zwanych stron. Strona jest po prostu bitmapą na której program rysuje pojedynczą klatkę animacji, a następnie wyświetla ją na ekranie.

BITMAP *create_video_bitmap(int width, int height); Alokuje pamięć video na bitmapy o określonym rozmiarze. Może być użyte do alokacji pozaekranowej pamięci video, aby zapisać gotową do wyświetlenia ramkę. Może być ona później wyświetlona przez operację sprzętowej zamiany wyświetlanych ramek, co można wykonać wywołując funkcję `show_video_bitmap()`.
Należy pamiętać o zwolnieniu ramki, zanim zostanie wywołana funkcja `set_gfx_mode()`.

void destroy_bitmap(BITMAP *bitmap); Zwalnia pamięć zarezerwowaną na daną bitmapę.

int show_video_bitmap(BITMAP *bitmap); Funkcja wyświetla zdefiniowaną bitmapę na ekranie, zastępując aktualnie wyświetlaną ramkę. Wstawiana bitmapa musi mieć taki sam rozmiar, jak ekran rzeczywisty (taki, jak wybrany tryb graficzny).

void clear_to_color(BITMAP *bitmap, int color); Czyści wskazaną bitmapę przy pomocy podanego koloru.

3.4. Obsługa klawiatury

Biblioteka `Allegro` posiada własny interfejs, który odpowiedzialny jest za obsługę komunikacji z użytkownikiem.

int keypressed(); Zwraca wartość `TRUE` jeżeli w buforze wejścia znajduje się wartość do odczytu. Funkcja umożliwia sprawdzenie, czy wywołanie kolejne funkcji `readkey()` spowoduje zablokowanie programu. Możliwe jest też wykorzystanie tej funkcji do wyświetlania animacji do czasu wciśnięcia dowolnego przycisku.

void clear_keybuf(); Umożliwia wyczyszczenie bufora danych wejściowych. Użycie tej funkcji może być przydatne, jeżeli dane wczytane przez funkcje `readkey()` lub `ureadkey()` powinny być aktualne i nie pochodzić od wcześniej wciśniętych przycisków.

extern volatile char key[KEY_MAX]; Tablica flag zawierająca informację,. Który z przycisków specjalnych jest aktualnie wciśnięty. Z tego powodu tablica ta jest jedynie do odczytu. Wartości w tej tablicy ustawiane są asynchronicznie, ale jeżeli funkcja `keyboard_needs_poll()` zwróci `TRUE`, należy ręcznie wywołać funkcję `poll_keyboard()`. Kody przycisków umieszczone są w nagłówku `allegro/keyboard.h` jako serie stałych zaczynających się od `KEY_*`. Niektóre przykłady zostały wypisane poniżej:

```
KEY_A ... KEY_Z,
KEY_0 ... KEY_9,
KEY_0_PAD ... KEY_9_PAD,
KEY_F1 ... KEY_F12,

KEY_ESC, KEY_TILDE, KEY_MINUS, KEY_EQUALS, KEY_BACKSPACE, KEY_TAB, KEY_OPENBRACE, KEY_CLOSEBRACE
KEY_ENTER, KEY_COLON, KEY_QUOTE, KEY_BACKSLASH, KEY_BACKSLASH2, KEY_COMMA, KEY_STOP,
KEY_SLASH, KEY_SPACE,

KEY_INSERT, KEY_DEL, KEY_HOME, KEY_END, KEY_PGUP, KEY_PGDN, KEY_LEFT, KEY_RIGHT, KEY_UP,
KEY_DOWN,

KEY_SLASH_PAD, KEY_ASTERISK, KEY_MINUS_PAD, KEY_PLUS_PAD, KEY_DEL_PAD, KEY_ENTER_PAD,

KEY_PRTSCR, KEY_PAUSE,

KEY_ABNT_C1, KEY_YEN, KEY_KANA, KEY_CONVERT, KEY_NOCONVERT, KEY_AT, KEY_CIRCUMFLEX,
KEY_COLON2, KEY_KANJI,

KEY_LSHIFT, KEY_RSHIFT, KEY_LCONTROL, KEY_RCONTROL, KEY_ALT, KEY_ALTGR, KEY_LWIN, KEY_RWIN,
KEY_MENU, KEY_SCRLOCK, KEY_NUMLOCK, KEY_CAPSLOCK

KEY_EQUALS_PAD, KEY_BACKQUOTE, KEY_SEMICOLON, KEY_COMMAND
```

Przykład:

```
if (key[KEY_SPACE])
printf("Wciśnięto spację\n");
```

3.5. Obsługa tekstu

W trybie graficznym, tekst nie może być wyświetlany przy pomocy funkcji `printf()` i wymagane jest wykorzystanie dedykowanych do tego funkcji biblioteki `Allegro`.

void textout_ex(BITMAP *bmp, const FONT *f, const char *s, int x, int y, int color, int bg); Funkcja wypisuje ciąg `s` na bitmapie `bmp` na pozycji `(x, y)` przy użyciu fontu `font` o kolorze `color` i tle koloru `bg`. Jeżeli kolor tła jest ustawiony na `-1`, to tekst jest wyświetlany przezroczyście. Jeżeli wartość `-1` jest podana jako kolor napisu, to użyty jest domyślny kolor fontu.

extern FONT *font; Prosta czcionka o stałym rozmiarze 8x8. W dokumentacji biblioteki można znaleźć opis, jak zmienić standardową czcionkę na własną.

void textout_centre_ex(BITMAP *bmp, const FONT *f, const char *s, int x, y, int color, int bg); Funkcja działa podobnie do `textout_ex()`, jednak koordynaty określają nie lewą krawędź napisu, ale jego środek. Funkcja przydaje się, gdy wyświetlany tekst ma zostać wyśrodkowany. Przykład:

```
texttt /* Important texts go in the middle. */
width = text_length("GAME OVER");
textout_centre_ex(screen, font, "GAME OVER",
SCREEN_W / 2, SCREEN_H / 2,
makecol(255, 0, 0), makecol(0, 0, 0));
```

4. Zadania

Uwaga! Wszystkie pogramy muszą być napisane z podziałem na funkcje z parametrami.

1. Napisz program, który umożliwi użytkownikowi poruszanie za pomocą klawiszy kursora okręgiem o promieniu 10 pikseli po ekranie. Program **nie powinien** umożliwiać opuszczenia ekranu przez okrąg.

2. Pobierz ze strony przedmiotu program symulujący ruch piłki po ekranie. Zmodyfikuj go dodając „paletkę" w postaci odcinka znajdującego się na dole ekranu, którą użytkownik będzie mógł poruszać za pomocą klawiszy kursora. Piłka powinna odbijać się od paletki, jeśli napotka ją na swojej drodze.