ISSN 0239–8044

1
1993

Techniki Komputerowe

BIULETYN INFORMACYJNY

INSTYTUT MASZYN MATEMATYCZNYCH
WARSZAWA 1993

Techniki Komputerowe

BIULETYN INFORMACYJNY

Rok XXVIII, Nr 1, 1993

INSTYTUT MASZYN MATEMATYCZNYCH
WARSZAWA 1993

Wydaje:

INSTYTUT MASZYN MATEMATYCZNYCH
UL. KRZYWICKIEGO 34
02-798 WARSZAWA
TEL. 21.84.41, TLX 81.78.80, FAX 29.92.70

Skład komputerowy: *ANDRZEJ ABRAMOWICZ*

Od wydawcy:

Szanowny Czytelniku,

Wznawiamy wydawanie Biuletynu. W ninieszym numerze, odmiennym w swojej szacie wydawniczej od poprzednich, zamieściliśmy artykuły związane tematycznie z pracami, którymi zajmuje się Instytut. Ponieważ pozostajemy w swojej działalności badawczo-rozwojowej w dziedzinie zastosowań informatyki, są tutaj informacje o narzędziach sprzętowych i programowych, a także o zagadnieniach normalizacyjnych. Podejmujemy próbę prezentacji własnych poglądów i opracowań w takiej formie, aby była ona dostępna dla wszystkich zainteresowanych informatyką.

TECHNIKI KOMPUTEROWE

Rok XXVIII Nr 1 1993

Spis treści

Str.

Jądro komputerowego systemu oceny poziomu wiedzy, Janusz Janowski, Wojciech Przyłuski 5

Modularny edytor tekstów, Marek Kotowski 15

Laserowe metody uzyskiwania obrazów przestrzennych w przemysłowych systemach widzenia maszynowego, Romuald Synak 27

Wirtualny system pomiarowy VIRT II, Jarosław Wójtowicz 39

Systemy wytwórcze do edukacji w komputerowo zintegrowanym wytwarzaniu (CIM), Andrzej Kaczmarczyk 43

Projekt nowych standardów dla zestawu znaków w przetwarzaniu i przesyłaniu informacji, Hanna Kuźnicka, Jan Ryżko 55

JANUSZ JANOWSKI
INSTYTUT MASZYN MATEMATYCZNYCH WARSZAWA
WOJCIECH PRZYŁUSKI
INSTYTUT MASZYN MATEMATYCZNYCH WARSZAWA

Jądro komputerowego systemu oceny poziomu wiedzy

The Shell of Tests for Estimation of Knowledge Level

Streszczenie

Zrealizowano prosty język tworzenia zbioru źródłowego testu. Oprogramowanie narzędziowe, Analizator i Interpretator, przekształca źródło w test komputerowy. Metodą tą można implementować wysoce personifikowane testy komputerowe.

Abstract

Simple language was created for writing source file of test. The programming tools, Checker and Interpreter, convert the source into computer test. By this means highly personificated computer tests can be implemented.

1. Wstęp

Prezentowane oprogramowanie narzędziowe powstało w Instytucie Maszyn Matematycznych i jest przeznaczone do budowania testów komputerowych służących testowaniu i uczeniu. Zostało ono zrealizowane na zamówienie Ministerstwa Zdrowia i Opieki Społecznej dla programu „Lekarz Domowy".

Test polega na wyborze odpowiedzi na kolejne problemy-pytania. Każdy wybór jest oceniany w różnych aspektach i oceny są kumulowane z dotychczasowymi ocenami łącznymi. Może on zostać skomentowany różnymi tekstami w zależności od aktualnych ocen, ilości pytań, na które udzielono odpowiedzi i sum wag tych pytań w każdym aspekcie. Od tych wartości może zależeć wybór następnego pytania lub zakończenie testu. W każdej chwili jest możliwe obejrzenie aktualnego stanu ocen i ewentualne przerwanie testu.

Konstrukcja testu polega na przygotowaniu tekstu (zbioru tekstowego zwanego źródłem testu) o prostej budowie, do napisania którego można użyć każdego edytora ASCII.

Narzędziowe oprogramowanie przekształca tak przygotowany test komputerowy w program, który modelując działanie rzeczywistego egzaminatora umożliwia ocenę poziomu wiedzy egzaminowanego — ogólną i wycinkową — w zakresie wybranych zagadnień.

Oprogramowanie narzędziowe testów składa się z dwóch programów:

- Analizatora, który służy do analizy poprawności źródła testu. Jeśli źródło testu nie zawiera błędu, to wynikiem analizy jest utworzenie dodatkowego zbioru dyskowego zawierającego adresy pewnych pozycji źródła testu.
- Interpretatora, który wykorzystując zanalizowane źródło testu oraz jego zbiór adresowy przeprowadza test z operatorem komputera.

Tworzenie źródła testu jest technicznie proste i polega na wypełnieniu odpowiedniego formularza treścią: pytań, odpowiedzi, komunikatów i przejść. Jednak wypełnienie pewnych pozycji tego formularza, np. dotyczących ważności pytań, ocen odpowiedzi, warunkowych przejść do następnego pytania i komunikatów, wymaga wiedzy merytorycznej i przyjęcia koncepcji egzaminowania (np. system oceniania jest niezadowalający, gdy losowe udzielanie odpowiedzi prowadzi do średniej oceny; system przejść do następnego pytania jest niezadowalający, jeśli możliwe jest zakończenie testu po udzieleniu odpowiedzi jedynie na łatwe pytania).

1.1. Jak pisać testy w prezentowanym systemie

Aby napisać test za pomocą opisywanego systemu, należy ustalić pewne istotne dotyczące go założenia.

Po pierwsze, trzeba ustalić, czy pytania zadawane testowanemu mają służyć do oceny jednego czy też kilku (dopuszcza się maks. 9) aspektów stanu wiedzy testowanego. Na przykład, można testując kogoś z dziedziny historii stawiać takie pytania (wieloaspektowe), że jedna odpowiedź na dane pytanie pozwala egzaminującemu przyjąć kilka opinii dotyczących stanu wiedzy testowanego, np. w aspektach: znajomość dat, znajomość postaci, zdolność logicznego myślenia itp.

Po drugie, należy ustalić sposób oceniania testowanego. Każdy z ewentualnych aspektów jest oceniany oddzielnie w indywidualnym, ustalonym dla tego aspektu przez piszącego test, zakresie ocen. Każdemu aspektowi przyporządkować należy pewien współczynnik (wagę), który pozwala wyróżnić aspekty pod względem ważności. Aspekty z większą wagą mają odpowiednio większy wpływ na końcową łączną ocenę testu. Łączna ocena testu jest sumą ocen wszystkich aspektów tego testu. Ocena w danym aspekcie powstaje jako suma ocen w tym aspekcie za poszczególne odpowiedzi, przy czym brane są pod uwagę wagi pytań dotyczące tego aspektu. Biorąc to pod uwagę, piszący test musi każdemu pytaniu przypisać zestaw wag odpowiadających ustalonym aspektom. W ten sposób określa on ważność danego pytania w odniesieniu do poszczególnych aspektów testu. Ważność, o której mowa, to oczywiście wpływ jaki dane pytanie ma na oceny w rozważanych aspektach testu. Wreszcie piszący test musi każdej odpowiedzi przypisać zestaw ocen odpowiadających kolejnym aspektom. Przypisując te oceny (z ustalonego wcześniej dla każdego aspektu zakresu) piszący określa oczywiście wpływ, jaki dana odpowiedź mieć będzie na oceny testu w poszczególnych aspektach. Może na przykład przyznawać oceny wyłącznie w kategoriach prawda-fałsz, ale może także zdecydować się na przyznawanie ocen pośrednich.

Po trzecie, piszący test musi zdecydować się na sposób egzaminowania testowanego i może tu wykorzystać wiele różnych mechanizmów przygotowanych do budowania testu. Jedną z zasadniczych decyzji, jakie podejmie piszący test, jest ustalenie oddzielnie dla każdego pytania zestawu odpowiedzi. Można tu stosować wiele rozwiązań. Odpowiedzi może być mało lub wiele (maks. 15). Na przykład, można dwie z odpowiedzi określić jako bardzo dobre, a wszystkie pozostałe jako

złe, czyli o znacznie mniejszej ocenie. Kolejna ważna decyzja w sprawie sposobu egzaminowania to wybór układu pytań tworzących test. Trzeba zdecydować, czy każdy egzaminowany odpowiadać ma na te same pytania w ustalonej kolejności, co odpowiada zwykłym (nie komputerowym) testom, czy też modelujemy działania rzeczywistego egzaminatora uzależniając kolejne pytania od dotychczasowego przebiegu testu. W tym drugim przypadku podstawową rolę odgrywają system numeracji pytań oraz reguły przejścia do następnego pytania. Numerując pytania (w systemie paragrafowym) można podzielić wszystkie pytania testu na różnorodne podgrupy tematyczne (merytoryczne) lub funkcjonalne (pytania wstępne, pytania końcowe). Reguły przejścia do następnego pytania mogą określać następne pytanie testu wykorzystując:

- ilość już udzielonych odpowiedzi,
- dotychczasowe oceny w różnych aspektach,
- łączną ocenę,
- sumę wag zadanych pytań w różnych aspektach,

przy czym nie musi tu być konkretnie podany numer następnego pytania, a tylko określenie następnego pytania jako losowo wybranego spośród jakiejś grupy pytań, czy też jako pytania o zbliżonym do podanego układzie wag.

Szczególnie ważne jest opracowanie komunikatów wyświetlanych w czasie testowania i komunikatu końcowego, wskazujących zarówno na braki w wiedzy, jak i na dobre rezultaty.

Tak więc piszący może tworzyć testy realizujące różne style egzaminowania od klasycznego testu kartkowego po badanie wiedzy użytkownika w sposób oryginalny.

Prawdopodobnie najlepszą metodą będzie stopniowe budowanie testu albo przez łączenie mniejszych testów poświęconych poddziedzinom, albo przez stopniowe dołączenie trudniejszych i bardziej szczegółowych pytań. Konstrukcji formalno-technicznej źródła testu jest poświęcona reszta opracowania.

2. Zbiór — źródło testu

Zbiór ten musi być zgodny z ustaloną składnią.

Składnia określa kolejność i znaczenie poszczególnych informacji. Ogranicznikami informacji są nazwy zaczynające się od znaku „@": `@tytul`, `@oceny`, `@pytanie`, `@odp`, `@gdy`, `@koniec`. Rozpoczynają one opisy jednostek informacji, które muszą wystąpić w źródle testu w następującej kolejności:

`@tytul` (tekst tytułu)
`@oceny` (parametry systemu ocen)
`@pytanie` (informacje dotyczące pytania, wszystkie odpowiedzi na to pytanie, warunki określenia następnego pytania lub zakończenie testu)

⋮

`@pytanie` (...)
`@koniec`

Zanalizowane źródło testu nie zawierające błędów ma wpisaną na końcu sumę kontrolną i datę. Tylko takie źródło testu może być interpretowane.

Interpretacja zaczyna się zawsze od postawienia pierwszego pytania z w/w ciągu pytań. Ilość pytań nie może być większa niż 360.

Omówimy teraz pewne elementy konstrukcji źródła tekstu.

2.1. Komentarz

W źródle testu, tekst między dwoma kolejnymi „\" (włącznie z nimi) nie ma znaczenia. Oznacza to, że komentarz nie jest separatorem i nie może być zawarty jeden w drugim. Liczy się jednak jego długość, tzn. wydłuża linie w bloku tekstu i nie zmienia położenia pierwszego cudzysłowu bloku tekstu, gdy jest przed nim umieszczony.

Przykład

`12\zl\34` jest liczbą 1234.
Tekst „`12\zl\34`" ma szerokość 8 znaków.

2.2. Blok tekstu

Podstawowe informacje dla użytkownika, takie jak: tytuł testu, pytania, odpowiedzi, komunikaty i wydruki są blokami tekstu wyświetlanymi w odpowiednich oknach.

Blok tekstu mieści się między dwoma podwójnymi cudzysłowami. Ma on szerokość liczoną od jego pierwszego cudzysłowu (wyłącznie) do najdalej sięgającego w prawo znaku (włącznie) lub cudzysłowu (wyłącznie). Będzie wyświetlany w oknie o tej szerokości.

Długość jego wyznacza długość okna. Gdy przekracza ona 22 linie, tekst będzie przewijany. W wierszu bloku tekstu, do kolumny pierwszego cudzysłowu włącznie, mogą występować jedynie spacje lub komentarz.

Żaden tekst nie może przekroczyć 4095 znaków.

2.3. Tytuł

Tytuł jest blokiem tekstu, który jest lokalizowany w środku ekranu.

Przykład

```
@tytul "
                Lekarz Domowy
                  Kurs I
                                "
```

2.4. System ocen

Ocena testu posiada jednostkę wartości i różne aspekty.

Każdy aspekt ma własną jednostkę i zakres wartości.

Wartość oceny testu jest sumą wartości jej aspektów wyrażonych w jednostkach oceny testu.

Współczynnikami przeliczającymi są stosunki jednostek aspektów oceny do jednostki oceny testu, zwane dalej wagami (jednostek) aspektów.

Pytania testu mają wagę pytania dla każdego aspektu, a odpowiedzi — oceny we wszystkich aspektach.

Aktualna wartość aspektu oceny jest średnią ocen udzielonych odpowiedzi ważonych wagami pytań i znormalizowanych sumą tych wag.

W źródle testu system ocen jest określony przez podanie wagi i zakresu ocen dla każdego aspektu.

Przykład

```
@oceny  \Ocena testu jest procent trafnych odpowiedzi\
1       \aspekt pierwszy, jednostka = trafna_odpowiedz\
100     \waga aspektu pierwszego =
        trafna_odpowiedz/(procent*trafna_odpowiedz)\
(0,1)   \zakres aspektu pierwszego\
\Wszystkie pytania maja wage 1. Odpowiedzi trafne maja ocene 1, a
nietrafne ocene 0.\
```

Przykład

```
@oceny  \Ocena testu jest "kombinowany" procent trafnych odpowiedzi\
1       \aspekt pierwszy, jednostka = 0.75*trafna_odpowiedz\
75      \waga aspektu pierwszego =
        0.75*trafna_odpowiedz/(procent*trafna_odpowiedz)\
(0,1)   \zakres aspektu pierwszego\
2       \aspekt drugi, jednostka = 0.25*trafna_odpowiedz\
25      \waga aspektu drugiego =
        0.25*trafna_odpowiedz/(procent*trafna_odpowiedz)\
(0,1)   \zakres aspektu drugiego\
\Wszystkie pytania maja wagi 1 obu aspektow. Odpowiedzi trafne w
aspekcie mają ocene 1, a nietrafne ocene 0 w tym aspekcie.\
```

W wariantach powyższych przykładów możemy zaakcentować ważność pewnych pytań w jakimś aspekcie przez zwiększenie wagi pytania lub nadać odpowiedzi ocenę ułamkową stosownie do stopnia trafności.

Aktualny stan egzaminu jest dostępny w źródle poprzez opisane tu symbole. Zostaną one zamienione na wartości w tekstach komunikatów i warunkach wyboru następnego pytania (natomiast pozostaną symbolami w tekstach pytań i odpowiedzi).

Symbole `$1` ... `$9` oznaczają aktualną ocenę w aspektach 1...9 wyrażoną w jednostkach oceny testu.

Symbol `$0` oznacza aktualną ocenę testu (`$0=$1+...+$9`).

Symbol `#0` oznacza ilość udzielonych odpowiedzi.

Symbole `#1`...`#9` oznaczają sumy wag w poszczególnych aspektach wszystkich pytań, na które udzielono odpowiedzi. Wskazują one na dotychczasowy stopień przeegzaminowania testowanego w poszczególnych aspektach.

2.5. Pytania

Pytania mają:

- Numer
  zaczynający się od znaku „#" i będący dalej ciągiem cyfr i kropek, np. `#1.12.4`.
- Wagi
  będące układem nieujemnych liczb w nawiasach zwyczajnych, np. `(1 2)`. Stosunek wagi pytania w danym aspekcie do wag innych pytań w tym aspekcie wpływa na wartość oceny tego aspektu. Natomiast stosunek wagi do wag tego pytania w innych aspektach określa stopień równomierności przeegzaminowania w różnych aspektach.

- Blok tekstu pytania
  (wyświetlany w lewym górnym rogu ekranu).

  Przykład (łączny)

```
@pytanie #1.12.4
(1 2) \wagi\
"blok          \20x3\
   tekstu
      pytania."
```

- Ciąg nie więcej niż 16 odpowiedzi (lub pusty), z których każda zaczyna się od ogranicznika „@odp".

  Odpowiedzi mają:

- Blok tekstu odpowiedzi
  lokowany w prawym dolnym rogu ekranu.
- Układ ocen
  w każdym aspekcie w nawiasach zwykłych.
- Opcjonalny komunikat.

  Przykład (łączny)

```
@odp "blok tekstu odpowiedzi."
(0.5 1) \oceny odpowiedzi\
"Pierwsza opcjonalna czesc bloku tekstu komunikatu o wybranej
odpowiedzi, wynikach ogolnych $0 i ewentualnie w aspektach $1,
$2, $3, $4, $5, $6, $7, $8, $9 ilosci zadanych pytan #0 i stopniu
przeegzaminowania w aspektach #1, #2, #3, #4, #5, #6, #7, #8, #9."
```

- Ciąg (być może pusty) warunkowych przejść do następnego pytania, z których każde zaczyna się od ogranicznika „@gdy" i *obligatoryjne*, bezwarunkowe przejście do następnego pytania. Pierwszy spełniony warunek tego ciągu wyznacza wybór informacji następnego pytania. Test zostaje zakończony, gdy ciąg ten jest pusty. Teksty warunków mogą zajmować kilka linii. W warunkach mogą występować symbole $0...$9, #0...#9, liczby *bez* znaków, zwykłe nawiasy, operatory arytmetyczne +, −, *, /, symbole relacji <, >, = (relacje *muszą* być zamknięte w nawiasach), operatory logiczne v, &, ¯, w normalnej kolejności ważności. Reguła przejścia do następnego pytania może zawierać komunikat, który staje się dalszym ciągiem komunikatu odpowiedzi (razem nie więcej niż 22 linie) i jeden z trzech rodzajów wyboru następnego pytania (lub puste miejsce przy zakończeniu testu):
  1. Numer konkretnego pytania lub, gdy numer jest zakończony kropką, numer grupy pytań, z której będzie losowo wybrane następne pytanie.
  2. Ciąg do 10-ciu liczb przedzielonych pojedynczymi numerami pytań lub paragrafami. Losowy wybór jednej z tych liczb określa sąsiedni numer pytania lub paragraf, lub kończy test.
  3. Układ wag, według którego wybiera się pytanie o najbardziej podobnym układzie wag (w sensie bliskości do najmniejszej sumy absolutnych różnic).

Przykład (przesadnie komentowany, z wartościami liczbowymi dostosowanymi do wariantu drugiego przykładu systemu ocen)

```
@gdy
\wagi zadanych pytan w obu aspektach > 20 i ocena
>80\(#1>20)&(#2>20)&($0>80): "Dziekuje. Dobry wynik."
@gdy
\wagi zadanych pytan w obu aspektach > 20 i ocena >40
(<=80)\(#1>20)&(#2>20)&($0>40): "Dziekuje.Wynik mierny."
@gdy
\wagi zadanych pytan w obu aspektach > 20 (i ocena <=
40)\(#1>20)&(#2>20): "Dziekuje. Slaby wynik $0\xx\."
@gdy
\ilosc zadanych pytan >20 lub ilosc zadanych pytan >15 i ocena <40 lub
>80\(#0>20)v(#0>15)&(($0<40)v($0>80)): "Dziekuje.Dosc pytan."
\kontynuujemy test gdy: nie osiagnelismy koniecznego stopnia
przeegzaminowania obu aspektow i nie przekroczylismy limitu pytan i
jest sens zadawac dalsze pytania. Bedziemy zadawac trudne pytania w
pierwszej fazie testu lub gdy osiagane sa dobre wyniki.\
@gdy
\(ilosc zadanych pytan <10 lub ocena >60) i stopien przeegzaminowania w
aspekcie 1 jest mniejszy niz 0.6 stopnia przeegzaminowania w aspekcie 2
\((#0<10)v($0>60))&(#1<0.6*#2): "Pytanie trudne w aspekcie 1."  (3 0)
@gdy
\(ilosc zadanych pytan <10 lub ocena >60) i stopien przeegzaminowania w
aspekcie 2 jest mniejszy niz 0.6 stopnia przeegzaminowania w aspekcie 1
\((#0<10)v($0>60))&(#2<0.6*#1): "Pytanie trudne w aspekcie 2."  (0 3)
@gdy
\(ilosc zadanych pytan <10 lub ocena >60) (i stopien przeegzaminowania
w obu aspektach nie jest zbyt rozbiezny.)\(#0<10)v($0>60): "Pytanie
trudne w obu aspektach."  (3 3)
\W pozostalych przypadkach losowo wybieramy pytania latwe.\
: 0.2 #0.5          \konkretne pytanie\
  0.2 #1. (0 0)     \pytanie z paragrafu 1\
  0.2 #2. (0 0)     \pytanie z paragrafu 2\
  0.2 (0 0)         \pytanie z calego tekstu\
  0.2               \koniec\
```

2.6. Alias

W realnych zródłach testu należy się spodziewać dużej powtarzalności jego fragmentów np. komunikatów, warunków przejścia do następnego pytania lub warunków zakończenia testu. Współczesne edytory łagodzą ten problem przez łatwość kopiowania, tym niemniej nadmierna ilość tekstu będzie utrudniać merytoryczną analizę źródła. Dlatego wprowadzamy mechanizm zastępowania fragmentów źródła ich krótkimi opisami będącymi nazwami alias.

Alias są nazwami fragmentów tekstu źródła testu. Występują one w tekście w miejsce tych fragmentów. Analizator sprawdza formalną poprawność źródła testu po zastąpieniu alias ich treścią.

Są dwa rodzaje alias. Pierwszy może wystąpić w tekście źródła wyłącznie po swojej definicji. Drugi nie musi spełniać tego warunku. Jest to użyteczna własność, ale stwarzająca niebezpieczeństwo zapętlenia programów. W związku z tym wprowadzone jest ograniczenie zagnieżdżenia alias w ich definicjach (do 20 poziomów).

Przykład (opracowany na podstawie poprzedniego przykładu)

```
{wyjscie:
    @gdy (#1>20)&(#2>20)&($0>80): "Dziekuje. Dobry wynik."
    @gdy (#1>20)&(#2>20)&($0>40): "Dziekuje. Wynik mierny."
    @gdy (#1>20)&(#2>20): "Dziekuje. Slaby wynik $0\xx\."
    @gdy (#0>20)v(#0>15)&(($0<40)v($0>80)): "Dziekuje. Dosc pytan."}
{nastepne_pytanie:
    @gdy ((#0<10)v($0>60))&(#1<0.6*#2): "Pytanie trudne w aspekcie 1 z
    paragrafu 2." #2. (3 0)
    @gdy ((#0<10)v($0>60))&(#2<0.6*#1): "Pytanie trudne w aspekcie 2 z
    paragrafu 1." #1. (0 3)
    @gdy (#0<10)v($0>60): "Pytanie trudne w obu aspektach.
    "{pytanie_trudne |{pytania_latwe |}
```

Powyższe definicje będą obowiązywały we wszystkich pytaniach, jeśli zostaną umieszczone przed pierwszym pytaniem. Poprzedni przykład zapisze się wtedy następująco:

```
{wyjscie}
{pytanie_trudne: (3 3)}
{pytania_latwe:
:0.2 #0.5 0.2 #1. (0 0) 0.2 #2. (0 0) 0.2 (0 0) 0.2}
{nastepne_pytanie}
```

a jego wariant może mieć postać:

```
{wyjscie}
{pytanie_trudne:#1.0}{pytania_latwe:(0 0)}
{nastepne_pytanie}
```

2.7. Koniec

Źródło testu musi kończyć się ogranicznikiem „`@koniec`". Tekst źródła po jego pierwszym wystąpieniu jest kasowany.

3. Przykład

Przedstawiamy poniżej wybrane fragmenty fikcyjnego testu ilustrujące niektóre wyżej opisane konstrukcje. Przypominamy, że tekst pomiędzy znakami „\" jest tylko komentarzem i może być pominięty.

```
@tytul
"              TYTUL TESTU                "
@oceny
    \aspekt          waga          zakres ocen\
       1              25              (1 5)        \aspekt pomocniczy\
       2              75              (1 5)        \aspekt podstawowy\
       ⋮

@pytanie #3.7     (1 3)            \W 1 aspekcie waga pytania ma wartosc
                                   1, w 2 aspekcie odpowiednio wartosc 3.
                                   Wagi te ozn. skale trudnosci pytania w
                                   poszczegolnych aspektach testu.\
"              TRESC PYTANIA              "
   @odp
   "                TRESC ODPOWIEDZI                "
   (5 2)                           \Ocena odpowiedzi: 5 w aspekcie 1 oraz
                                   2 w aspekcie 2.\
       @gdy (#0>10):               \Jesli zostalo zadanych wiecej niz 10
                                   pytan to koniec testu.\
       @gdy ($1<2.5): #2.          \Jesli dotychczasowa ocena w 1
                                   aspekcie jest mniejsza niz 2.5 to
                                   przechodzimy do losowo wybranego
                                   pytania z grupy 2. .\
       @gdy ($2<2.5): #1.          \Jesli dotychczasowa ocena w 2
                                   aspekcie jest mniejsza niz 2.5 to
                                   przechodzimy do losowo wybranego
                                   pytania z grupy 1. .\
       @gdy ($0>4): #4. (5 5)      \Jesli dotychczasowa laczna ocena
                                   jest wieksza od 4 to przechodzimy do
                                   trudnego w obu aspektach pytania z
                                   grupy 4.\
                    : #3.8         \W pozostalych przypadkach
                                   przechodzimy do pytania o numerze 3.8
                                   .\
   @odp
   "                TRESC ODPOWIEDZI                "
   (1 1)                           \Ocena odpowiedzi: 1 w aspekcie 1 oraz
                                   1 w aspekcie 2.\
       @gdy (#0>10)&((#1>15)v (#2>15)):
                                   \Jesli zostalo zadanych wiecej
                                   niz 10 pytan oraz jesli w jednym z
                                   aspektow zadano pytania o sumie wag
                                   przekraczajacej 15 wtedy konczy sie
                                   test.\
```

```
©gdy ($0>3): (4 4)        \Jesli dotychczasowa laczna ocena
                          jest wieksza od 3 to przechodzimy do
                          pytania o ukladzie wag najbardziej
                          zblizonym do (4 4).\
             : 0.1 #5.1
               0.1 #5.2
               0.1 #6.1
               0.3
               0.4 #7.    \W pozostalych przypadkach
                          przechodzimy losowo
                          z prawdopodobienstwem 0.1 do
                          pytan o podanych numerach lub
                          z prawdopodobiebstwem 0.3 konczymy
                          test lub z prawdopodobienstwem 0.4
                          przechodzimy do jakiegos pytania
                          paragrafu 7.\
  :
©koniec
```

MAREK KOTOWSKI
INSTYTUT MASZYN MATEMATYCZNYCH WARSZAWA

Modularny edytor tekstów

Modular Text Editor

Streszczenie

W niniejszym opracowaniu przedstawiono projekt modularnego edytora tekstu. Opisana została maszyna edytorowa i jej funkcje elementarne. Przedstawiono również, w jaki sposób za pomocą funkcji elementarnych mogą być realizowane operacje edycji tekstu. Przedyskutowano zalety i wady koncepcji.

Abstract

The paper describes a concept of modular text editor. So called editor machine and its elementary functions are introduced. It is described, how text editing operations can be realized using elementary functions. Merits and drawbacks of the concept are also discussed.

1. Wstęp

Funkcje, zmienne i struktury podane w tekście są zdefiniowane w języku C (sam projekt edytora modularnego jest niezależny od systemu operacyjnego i języka programowania). Przyjęto przy tym zasadę stosowaną w [1]: identyfikatory, a zwłaszcza nazwy funkcji i typów, pisane są w języku angielskim, a komentarze — w języku polskim.

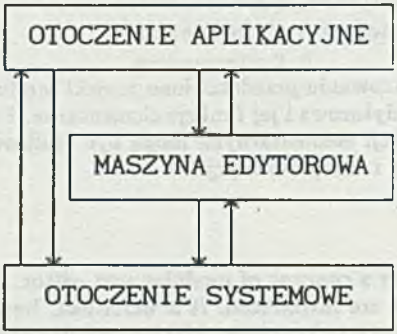
2. Koncepcja edytora modularnego

Program modularny jest to — najogólniej rzecz biorąc — program składający się z wydzielonych i w dużej mierze niezależnych jednostek (modułów), które mogą być łączone ze sobą w bardziej złożone konstrukcje; więcej: mogą być także łączone z innymi jednostkami, tworząc oddzielne produkty. W odniesieniu do omawianego edytora istota tej koncepcji polega na wydzieleniu funkcji edycyjnych w postaci tzw. maszyny edytorowej, niezależnej od aplikacyjnej części edytora i otoczenia systemowego. Maszyna edytorowa realizuje elementarne funkcje edycyjne (zwane dalej po prostu funkcjami elementarnymi) tak dobrane, by każda operacja edycji na tekście mogła dać się w nich zapisać. Termin „maszyna" jest tu najzupełniej odpowiedni: maszyna edytorowa pełni podobną rolę w całym edytorze tekstu, co np. silnik w samochodzie.

Edytor tekstu składa się z maszyny edytorowej oraz otoczenia systemowego i aplikacyjnego.

Otoczenie systemowe jest to zestaw funkcji realizujących odwołania do zasobów systemu, w tym m.in. operacje na urządzeniach wejścia-wyjścia, alokację pamięci itp. Wydzielenie otoczenia systemowego jako oddzielnej jednostki modularnej pozwala tworzyć edytory pracujące w różnych systemach operacyjnych bez wymiany innych modułów edytora.

Otoczenie aplikacyjne jest to zestaw funkcji, które realizują interfejs użytkownika. Są to m.in. funkcje obsługi menu, funkcje formatujące, funkcje definiujące zestawy znaków drukarskich itp. Otoczenie aplikacyjne tworzy jednocześnie interfejs z maszyną edytorową. Zakłada się, że elementarne funkcje maszyny edytorowej są tak dobrane, by za pomocą ich sekwencji dały się zapisać wszystkie operacje edycji edytora.



Rys. 1. Ogólny schemat modularnego edytora tekstu

Z założenia, bezpośredni dostęp do tekstu mogą mieć jedynie funkcje maszyny edytorowej.

3. Maszyna edytorowa

Maszyna edytorowa działa na jednostce tekstu, za którą przyjęto jeden akapit (patrz niżej). Jest to wybór najrozsądniejszy, ponieważ akapit jest pojęciem występującym w każdym tekście, formatowanym i znakowym (tzw. tekście ASCII). Akapit jest przy tym niezależny od formatu tekstu, w przeciwieństwie np. do linii tekstu, której szerokość określona jest szerokością tekstu lub akapitu. Tym niemniej przyjęcie akapitu za jednostkę tekstu nie zmniejsza ogólności rozumowania (patrz p. 5.1).

Maszyna edytorowa składa się z tzw. pola bieżącego oraz ze zbioru funkcji elementarnych, które — najogólniej rzecz biorąc — pozwalają przesyłać na to pole poszczególne jednostki tekstu (akapity) i modyfikować je. Oprócz tego z polem bieżącym i edycją jego zawartości związane są pewne zmienne (są one zdefiniowane niżej). Będziemy nazywać je zmiennymi maszyny edytorowej.

W następnych punktach opisane jest szerzej pojęcie akapitu oraz funkcje elementarne.

3.1. Akapity

Podstawową jednostką przetwarzanego tekstu jest akapit, czyli uporządkowany ciąg znaków. Wszystkie akapity są równouprawnione, niezależnie od tego, czy są puste (nie zawierają żadnych znaków), czy też są bardzo duże. Tekst w obrębie akapitu jest jednorodny, tzn. traktowany jest jako ciąg znaków bez znaków wyróżnionych. Nie ma a priori ograniczenia ani na wielkość akapitów, ani na ich liczbę.

Tekst jest to uporządkowany ciąg kolejnych akapitów. Zakładamy tym samym, że zdefiniowane są pojęcia pierwszego, ostatniego, następnego i poprzedniego akapitu. Nie zakłada się nic o wewnętrznej organizacji całego tekstu i sposobie wzajemnego powiązania akapitów.

Wśród wszystkich akapitów tekstu przetwarzanego zawsze jeden jest akapitem bieżącym, tj. takim, który znajduje się w polu bieżącym maszyny edytorowej. Na akapicie bieżącym (ściślej: na zawartości pola bieżącego) zdefiniowane są operacje elementarne. Aby dokonać zmiany w akapicie, należy uczynić go akapitem bieżącym (wprowadzić do pola bieżącego) i posłużyć się funkcjami elementarnymi. Pole bieżące może być po prostu jednorodnie adresowanym obszarem pamięci operacyjnej (przy organizacji pamięci w komputerach klasy IBM PC taki obszar może mieć rozmiar do 64 Kb). Przy akapitach większych niż dostępny jednorodny obszar pamięci operacyjnej, akapit bieżący może być np. przechowywany częściowo w dwóch plikach roboczych zawierających początkową i końcową część akapitu bieżącego (tzw. bufory wahadłowe). Rozmiar tak implementowanego pola bieżącego — a tym samym i maksymalna wielkość akapitu bieżącego — jest ograniczony dostępną pamięcią dyskową.

3.2. Zmienne maszyny edytorowej

Z akapitem bieżącym związana jest wartość przesunięcia bieżącego. Jest to miejsce w akapicie bieżącym, w które „patrzy" maszyna edytorowa. Odpowiada ono położeniu kursora w edytorze tekstu. Każdy znak wstawiany do akapitu bieżącego jest wstawiany w miejsce określone przez wartość przesunięcia bieżącego (podobnie z tego miejsca jest usuwany znak, jeśli w edytorze np. realizowana jest operacja usuwania znaku znajdującego się na pozycji kursora).

Drugą zmienną istotną dla funkcji elementarnych jest znacznik modyfikacji. Jeśli ma on wartość 1, oznacza to, że akapit bieżący był modyfikowany (wartość 0 tego znacznika oznacza, że akapit nie był modyfikowany).

W dalszych rozważaniach będą potrzebne nam definicje typów: `PAR` (akapit), `PAR_ADDRESS` (adres akapitu), `OFFSET` (wartość przesunięcia w akapicie) i `CHAR_ADDRESS` (adres znaku). Jakkolwiek z punktu widzenia całości ich szczegółowa struktura nie ma znaczenia, dla jasności wykładu przyjmiemy pewne założenia. I tak adres akapitu może być adresem elementu w łańcuchu połączonych dwustronnie akapitów:

```
typedef struct par
{
    char *par_text;          /* ADRES TEKSTU AKAPITU */
    struct par *next_par;    /* ADRES NASTĘPNEJ STRUKTURY
                             AKAPITU */
```

```
    struct par *previous_par;    /* ADRES POPRZEDNIEJ STRUKTURY
                                    AKAPITU */
}   PAR, *PAR_ADDRESS;
```

Tekst może być również zorganizowany w tablicę wskaźników na obszary alokowane dla poszczególnych akapitów itd.

OFFSET może być po prostu liczbą całkowitą bez znaków:

```
typedef unsigned int OFFSET;
```

Dla operacji wstawiania, pisania i usuwania z akapitu ciągów znaków przydatna będzie nam definicja LENGTH. Możnaby ją zdefiniować poprzez typ OFFSET, ale — by nie ograniczać rozważań — zdefiniujemy ją oddzielnie:

```
typedef unsigned int LENGTH;
```

Mając definicję adresu akapitu i wartości przesunięcia, możemy zdefiniować adres znaku w tekście. Składa się on z adresu akapitu i wartości przesunięcia w obrębie akapitu:

```
typedef struct
{
    PAR_ADDRESS par_address;
    OFFSET par_offset;
}   CHAR_ADDRESS;
```

Zdefiniujemy również kilka zmiennych, które będą wykorzystywane i modyfikowane przez funkcje elementarne, w tym dwie wspomniane wyżej zmienne:

```
    char *current;              /* ADRES POLA BIEŻĄCEGO */
    int modified;               /* ZNACZNIK, CZY AKAPIT BIEŻĄCY BYŁ
                                   MODYFIKOWANY */
    OFFSET current_offset;      /* BIEŻĄCE PRZESUNIĘCIE W POLU /*
                                   BIEŻĄCYM */
    OFFSET par_length;          /* DŁUGOŚĆ AKAPITU */
    PAR_ADDRESS current_par;    /* ADRES AKAPITU BIEŻĄCEGO */
```

Konkretne typy tych zmiennych mogą być zdefiniowne w implementacji maszyny edytorowej.

3.3. Funkcje elementarne

Na akapicie bieżącym (w polu bieżącym) zdefiniowana jest pewna liczba elementarnych operacji edycyjnych. Są one podane poniżej jako funkcje w sensie języka C (ściślej: ich prototypy), przy czym nie opisujemy tu szczegółów działania tych funkcji (pewne szczegóły działania funkcji elementarnych podane są na końcu listy).

`Etake_current (PAR_ADDRESS)`

Pobiera dany akapit do pola bieżącego. Argumentem funkcji jest adres akapitu.

`Etake_next (void)`

Pobiera następny akapit do pola bieżącego.

`Etake_previous (void)`

Pobiera poprzedni akapit do pola bieżącego.

`Etake_first (void)`
Pobiera pierwszy akapit tekstu do pola bieżącego.

`Etake_last (void)`
Pobiera ostatni akapit tekstu do pola bieżącego.

`Eleave_current (void)`
Kopiuje bieżący akapit do pamięci (operacja kopiowania faktycznie realizowana jest tylko wtedy, gdy akapit był modyfikowany).

`Einsert_char (char)`
Wstawia znak w miejsce określone przez wartość bieżącego przesunięcia (tekst akapitu bieżącego jest rozsuwany). Argumentem funkcji jest wstawiany znak.

`Edelete_char (void)`
Usuwa znak z miejsca określonego przez bieżące przesunięcie (akapit bieżący jest kompresowany).

`Ewrite_char (char)`
Wpisuje znak w miejsce określone przez bieżące przesunięcie (znak znajdujący się w danym miejscu jest zamazywany).

`Einsert_string (char *, LENGTH)`
Wstawia ciąg znaków do akapitu bieżącego w miejsce określone przez bieżące przesunięcie (znaki znajdujące się w danym miejscu są rozsuwane). Argumentem funkcji jest adres ciągu, który ma być wstawiony, i jego długość.

`Edelete_string (LENGTH)`
Usuwa ciąg znaków z miejsca określonego przez bieżące przesunięcie (akapit bieżący jest kompresowany). Argumentem funkcji jest długość ciągu do usunięcia.

`Ewrite_string (char *, LENGTH)`
Wpisuje ciąg znaków do akapitu bieżącego w miejsce określone przez bieżące przesunięcie (znaki znajdujące się w danym miejscu są zamazywane). Argumentem funkcji jest adres ciągu, który ma być wstawiony, i jego długość.

`Eset_current_offset (OFFSET)`
Ustawia bieżące przesunięcie w polu bieżącym. Argumentem funkcji jest wartość przesunięcia w obrębie bieżącego akapitu.

`Eincrement_offset (void)`
Zwiększa wartość przesunięcia bieżącego o 1.

`Edecrement_offset (void)`
Zmniejsza wartość przesunięcia bieżącego o 1.

`Einsert_par (void)`
Wstawia nowy akapit w miejscu określonym przez wartość przesunięcia bieżącego (funkcja ta realizuje dzielenie akapitu bieżącego na dwa oddzielne akapity).

`Edelete_par (void)`
Usuwa akapit (funkcja ta łączy akapit bieżący z następnym w jeden akapit).

Dodatkowe funkcje, o jakie można wzbogacić plik funkcji elementarnych, opisane są niżej (p. 4.7).

Funkcje elementarne zmieniają wartości poszczególnych zmiennych maszyny edytorowej. I tak funkcje pobierające akapity do pola bieżącego nadają wartości zmiennym `current_par` i `par_length`. Funkcje zmiany wartości bieżącego przesunięcia zmieniają wartość zmiennej `current_offset`. Podobnie funkcje modyfikujące zawartość akapitu bieżącego nadają wartość 1 znacznikowi `modified`, a także zmieniają wartość zmiennej `par_length`.

Zdefiniowane wyżej funkcje elementarne nie są oczywiście całkowicie rozłączne. Np. funkcję `Eincrement_offset` można także zapisać jako funkcję `Eset_current_offset (current_offset+1)`. Podobnie jest z funkcjami `Ewrite_string` i `Ewrite_char`, `Einsert_string` i `Einsert_char` oraz `Edelete_string` i `Edelete_char`. Tym niemniej takie zdefiniowanie funkcji jest wygodne. Np. wspomniana funkcja `Eincrement_offset` będzie używana wielokrotnie, zwłaszcza przy wprowadzaniu znaków czy przesuwaniu kursora w lewo lub w prawo o jeden znak (oczywiście w konkretnej implementacji można liczbę funkcji elementarnych zredukować, nie zmieniając ogólnej koncepcji).

Podane definicje funkcji elementarnych są bardzo ogólne. Nie zakładamy nic o sposobie wewnętrznego działania funkcji elementarnych. Jeśli przyjmiemy opisany łańcuchowy model orgnizacji tekstu, to np. funkcja `Etake_next` będzie wprowadzała do pola pamięci akapit, którego adresem jest wyrażenie `current_par->next_par` (funkcja `Etake_previous` — odpowiednio `current_par->previous_par`). Można również przyjąć, że funkcja `Etake_next` nadaje przesunięciu bieżącemu domyślnie wartość 0, a funkcja `Etake_previous` nadaje temu przesunięciu wartość równą długości pobranego akapitu (odpowiada to w pierwszym przypadku ustawieniu kursora na początku akapitu, w drugim — na końcu). Podobnie w różny sposób mogą działać inne funkcje, np. `Einsert_char`, która może automatycznie zwiększać wartość przesunięcia bieżącego o 1 (jest to sprawa implementacji) i funkcja `Etake_current`, której argumentem może być pełny adres znaku (funkcja oprócz pobrania akapitu na pole bieżące będzie również nadawać odpowiednią wartość przesunięciu bieżącemu).

3.4. Modyfikowanie akapitu

Jak wspomniano wyżej, przy ładowaniu nowego akapitu do pola bieżącego znacznikowi `modified` jest nadawana wartość 0 (realizują to wszystkie funkcje pobierania akapitów). Każda z funkcji modyfikujących akapit bieżący nadaje znacznikowi `modified` wartość 1 (funkcje wstawiające i usuwające akapity również, przy czym tu sytuacja może być bardziej skomplikowana, zależnie od implementacji operacji dzielenia akapitu). Na tej podstawie funkcje zmieniające bieżący akapit na inny mogą dokonać ewentualnych zmian w tekście (np. alokować nowy obszar dla akapitu, przepisać do niego nową treść akapitu i zwolnić stary obszar). Może to także realizować sama funkcja `Eleave_current`, którą otoczenie aplikacyjne wykorzystuje przed każdą zmianą akapitu bieżącego (przynajmniej w tym celu została ona wprowadzona). Jest ona szczególnie przydatna w sytuacji, gdy trzeba porzucić edycję akapitu, nie ładując do pola bieżącego innych akapitów tego tekstu, np. przy zmianie jednego tekstu na inny.

4. Otoczenie aplikacyjne

Poniżej opisane są wybrane operacje edycji i sposób ich realizacji za pomocą funkcji elementarnych maszyny edytorowej.

4.1. Format tekstu

Pojęcie formatu tekstu jest związane z jego wizualizacją. Ten sam tekst może być wyświetlony lub wydrukowany z szerokością 60 znaków, a kiedy indziej — z szerokością 80 znaków (format może zresztą być podany w jednostkach długości). Podobnie rzecz się ma z wyrównaniem i z innymi elementami formatowania tekstu.

Maszyna edytorowa nie formatuje tekstu. Jest on formatowany przez funkcje otoczenia aplikacyjnego dopiero w momencie wyprowadzania go na urządzenie wyjściowe, takie jak drukarka czy monitor ekranowy. Dzięki temu oszczędza się znacznie czas przy operacjach zmian formatu większej ilości tekstu — formatowanych jest dokładnie tyle akapitów, ile akurat jest wyświetlonych na ekranie.

Sam format całego tekstu jest definiowany w otoczeniu aplikacyjnym, a format własny akapitu np. przez poszerzenie struktury `PAR` o opis formatu.

4.2. Realizacja poleceń edytora

Wszystkie operacje realizowane przez użytkownika w edytorze tekstu z założenia składają się

- z sekwencji wywoływanych elementarnych funkcji maszyny edytorowej,
- z operacji wizualizacji zmian, jeśli są one potrzebne (w tym operacji formatowania).

Nie ma żadnych założeń co do kolejności operacji modyfikacji tekstu i wizualizacji zmian. W dokonanej implementacji maszyny edytorowej (patrz p. 5.3) przyjęto, że funkcje elementarne na bieżącym akapicie są pierwotne w stosunku do operacji wizualizacji zmian. Innymi słowy: najpierw będzie modyfikowany sam tekst akapitu bieżącego w polu bieżącym, a dopiero potem skutek modyfikacji wyświetlany na ekranie. Operacje te można zrealizować w kolejności odwrotnej, np. przy wstawianiu znaków do tekstu znaki te wyświetlane są najpierw na ekranie, a dopiero potem następuje wstawienie ich do akapitu bieżącego, przeformatowanie akapitu i stosownie do wyniku skorygowanie zawartości ekranu.

4.3. Ruchy kursora

Zmiany położenia kursora edytora polegają na zmianie adresu znaku w tekście. Zmiana ta może się sprowadzać tylko do zwiększenia lub zmniejszenia wartości przesunięcia bieżącego w akapicie albo do zmiany akapitu bieżącego na inny. W pierwszym przypadku są to operacje:

```
Eincrement_offset();  /* PRZESUNIĘCIE KURSORA O 1 ZNAK W PRAWO */
Edecrement_offset();  /* PRZESUNIĘCIE KURSORA O 1 ZNAK W LEWO */
Eset_current_offset(current_offset +10);
                      /* PRZESUNIĘCIE KURSORA O 10 ZNAKÓW W PRAWO */
```

Jeśli przesunięcie kursora naprowadza go na początek następnego akapitu, należy wywołać funkcje:

```
Eleave_current();
```

```
Etake_next();
Eset_current_offset((OFFSET)0);
```

Podobnie polecenie przesunięcia tekstu o stronę (np. naciśnięcie klawisza [PgDn]) wymaga wywołania funkcji `Eleave_current`, a następnie cyklicznego pobierania kolejnych akapitów (`Etake_next`), formatowania ich, by obliczyć ile linii zawierają, aż do chwili, gdy będzie ich tyle, ile wynosi wielkość strony na ekranie (przykładowy proces przesuwania zawartości ekranu edytora jest opisany dokładniej w p. 4.4).

Podobnie można definiować markery, tj. wybrane miejsca w tekście. Definicja markera składa się po prostu z adresu znaku, z ewentualnym numerem markera. Przejście do markera sprowadza się do sekwencji:

```
Eleave_current();
Etake_current(marker_address);
Eset_current_offset(marker_offset_in_par);
```

Dwie ostatnie operacje elementarne da się zapisać krócej, jeśli przyjmie się wspomnianą wyżej alternatywną wersję funkcji `Etake_current`, z pełnym adresem znaku jako argumentem.

4.4. Wstawianie znaku

Operacja wstawiania znaku „c" do akapitu w miejsce określone przez wartość bieżącego przesunięcia polega zazwyczaj na realizacji funkcji elementarnych:

```
Einsert_char(c);
Eincrement_offset();
```

Z sytuacją taką mamy do czynienia gdy znak wprowadzany jest z klawiatury, tak jak pisze się na maszynie (kursor przesuwany jest o 1 znak w prawo). Otoczenie aplikacyjne jest przy tym odpowiedzialne za właściwe ustawienie kursora w edytorze. Np. w razie potrzeby przesuwa kursor do początku następnej linii (tej operacji maszyna edytorowa „nie widzi" — dla niej operacja przesuwania kursora sprowadza się jedynie do zwiększenia wartości przesunięcia bieżącego). Sytuacja może być jednak bardziej złożona. Otoczenie aplikacyjne jest odpowiedzialne także za przesuwanie tekstu na ekranie. Dla przykładu: jeśli wprowadzany jest znak na koniec linii znajdującej się na samym dole ekranu, a edytor w takiej sytuacji ma przesunąć zawartość ekranu o połowę linii w górę, musi wykonać całą sekwencję operacji. Oto przykładowa wersja takiej sekwencji:

- Porzucenie akapitu bieżącego (funkcja `Eleave_current`), który — jeśli został zmodyfikowany, zostanie zapisany w pamięci (otoczenie aplikacyjne musi przechować adres tego akapitu i wartość bieżącego przesunięcia w nim).
- Pobranie (funkcja `Etake_current`) na pole bieżące akapitu, od którego ma być wyświetlony tekst na ekranie. Jeśli zawartość ekranu ma być przesunięta o połowę w górę, to akapitem tym będzie akapit znajdujący się w połowie ekranu (np. 12 linii nad linią, gdzie znajduje się kursor). Akapit ten będzie teraz znajdował się w górnej linii ekranu).
- Ustawienie bieżącego przesunięcia (funkcja `Eset_current_offset`) na początek fragmentu tekstu akapitu, który ma być wyświetlony w górnej linii (może to być np. przesunięcie odpowiadające trzeciej linii akapitu, która stanie się pierwszą linią w oknie edycyjnym).

- Wyświetlenie całego ekranu (kolejne wywołania funkcji `Etake_next`, jeśli wyświetlanych będzie kilka akapitów, formatowanie akapitów i wyświetlanie linii tekstu na ekranie).
- Po wyświetleniu zawartości ekranu uczynienie bieżącym (`Etake_current`) akapitu, do którego były wprowadzane znaki (jego adres został przechowany przez funkcje otoczenia aplikacyjnego).

Przedstawiony model przesuwania zawartości ekranu (pomijamy tu zupełnie problem struktur danych wiążących ekran z akapitami) nie jest oczywiście jedynym możliwym. Ilustruje on natomiast sposoby wykorzystywania funkcji elementarnych maszyny edytorowej do realizowania takich operacji.

4.5. Usuwanie znaku

Operacja usuwania znaku z akapitu bieżącego z miejsca położenia kursora sprowadza się do wykonania funkcji `Edelete_char`, przy czym jeśli wartość bieżącego przesunięcia jest równa długości akapitu, usunięcie znaku równoważne jest usunięciu akapitu (akapit następny jest dołączany do bieżącego). Sytuację tę może rozpoznawać otoczenie aplikacyjne, a następnie wywoływać odpowiednie funkcje (może tę operację realizować również funkcja elementarna `Edelete_char`). Po połączeniu akapitów bieżące przesunięcie ma wartość taką, jaką miało w chwili realizowania operacji łączenia.

4.6. Zamiana ciągu znaków

Operacja zamiany w tekście ciągu znaków na inny polega na pobieraniu kolejnych akapitów do pola bieżącego (funkcje `Etake_next` lub `Etake_previous`, w zależności od kierunku realizacji operacji) oraz — po znalezieniu ciągu spełniającego kryteria podmiany — na realizacji funkcji `Edelete_string` (usunięcie ciągu zamienianego) i `Einsert_string` (wstawienie ciągu zamieniającego):

```
Etake_next();/Etake_previous();
...................
Edelete_string();
Einsert_string();
```

4.7. Operacje na blokach

Blok jest pojęciem zewnętrznym w stosunku do maszyny edytorowej. Oznacza to, że „nie widzi" ona bloków, jedynie kolejne akapity. Jeśli w tekście ma być zdefiniowany blok, można to zrobić na różne sposoby, np. podając adres pierwszego i ostatniego akapitu tworzącego blok albo wprowadzając wprost do tekstu specjalne znaki — lub sekwencje znaków — oznaczające początek i koniec bloku. Ale to rozwiązanie ma wadę: znaki te będą traktowane przez maszynę edytorową jako normalne znaki tekstu i tym samym ograniczą dopuszczalny zakres znaków samego tekstu (w istocie jest to sprawa otoczenia aplikacyjnego). Operacje usuwania i kopiowania bloków będą realizowane za pomocą funkcji elementarnych maszyny edytorowej: `Edelete_string`. `Einsert_string`, `Edelete_par`, `Einsert_par` itd.

UWAGA: Ten model realizacji operacji blokowych, jakkolwiek spójny, może być czasochłonny. Można z niego zrezygnować realizując w otoczeniu aplikacyjnym

pewne operacje nie korzystające z funkcji maszyny edytorowej (np. usuwanie łańcucha akapitów i „wiązanie" dwóch pozostałych części). Może to wydatnie skrócić czas realizacji operacji, ale oczywiście zmniejsza uniwersalność rozwiązania (część operacji edycyjnych nie jest realizowana przez maszynę edytorową, wiążąc otoczenie aplikacyjne bezpośrednio z tekstem). Lepszym rozwiązaniem jest rozszerzenie zbioru funkcji elementarnych maszyny edytorowej o operacje usuwania i wstawiania bloku (np. `Edelete_block` i `Einsert_block`).

Podobnie będą traktowane bloki drukarskie (ich interpretacja należy do otoczenia aplikacyjnego).

4.8. Inne operacje edycyjne

Wszystkie operacje edycyjne na tekście można z zasady zakodować za pomocą sekwencji funkcji elementarnych maszyny edytorowej. Przykładem może być proces weryfikacji poprawności ortograficznej tekstu. Funkcje weryfikacyjne pobierają kolejne akapity tekstu (`Etake_next` lub `Etake_previous`, zależnie od kierunku weryfikacji) wydzielają z akapitu bieżącego kolejne słowa i dokonują weryfikacji, sięgając do słownika ortograficznego. Poprawki w tekście wpowadzane są za pomocą funkcji elementarnych, tak jak w przypadku zwykłej edycji.

4.9. Praca z kilkoma tekstami

Maszyna edytorowa „nie wie" nic o tym, jaki tekst przetwarza. Można go zmienić na inny. Otoczenie aplikacyjne może przechowywać dowolną liczbę opisów tekstów, przy czym z każdym tekstem musi być związana informacja o jego akapicie bieżącym (zmienne opisane w p. 3.2), tj. o tym, w którym znajduje się kursor. Zmieniając adres akapitu bieżącego na adres odpowiedniego akapitu w innym tekście, otoczenie aplikacyjne może dowolnie zmieniać przetwarzane teksty.

5. Podsumowanie

5.1. Zalety rozwiązania

Opisane wyżej rozwiązanie posiada szereg zalet. Edytor można łatwo rozszerzyć o nowe opcje. Każde nowe polecenie sprowadza się do zakodowania sekwencji odpowiednich funkcji elementarnych. Struktura edytora jest modularna (funkcje wizualizacji tekstu i wprowadzanych zmian, pobierania rozkazów, wyświetlania menu, funkcje dyskowe itp. są niezależne od maszyny edytorowej). Dzięki temu tę samą maszynę edytorową można wykorzystać w dowolnym pakiecie przetwarzania tekstów. Trzeba jedynie zaprojektować i zrealizować odpowiednie funkcje aplikacyjne. Nawiązując do początkowego porównania maszyny edytorowej z silnikiem samochodowym można powiedzieć, że nowe otoczenia aplikacyjne są po prostu nowymi środkami lokomocji, w których można zastosować ten sam silnik (samochód z inną karoserią, jacht motorowy itp.). Można też stosować ograniczone wersje maszyny edytorowej. Np. dla programu przeglądania tekstu maszyna edytorowa może zawierać tylko funkcje zmieniające bieżący akapit (`Etake_first`, `Etake_last`, `Etake_next`, `Etake_previous`). Innym przykładem mogą być systemy tworzenia i obsługi słowników, encyklopedii, informacji prasowych itp. W takich przypadkach jednostką przetwarzania słownika lub encyklopedii nie będzie

akapit, a np. hasło złożone z kilku akapitów (samo hasło, jego opis, odnośniki, literatura itp.) i opisane w odpowiedni sposób (otoczenie aplikacyjne może dla jednego hasła realizować edycję np. czterech kolejnych akapitów).

Formatowanie tekstu jest procesem dynamicznym i ograniczonym do obszaru tekstu, który jest wyprowadzany na urządzenie wyjściowe (do przeformatowania całego tekstu wystarczy przeformatować tylko tyle tekstu, ile mieści się na ekranie).

Dzięki modularnej strukturze edytora można uzyskiwać — w systemach operacyjnych, które na to pozwalają, np. OS/2 — równoległość przetwarzania kilku tekstów, za pomocą jednej maszyny edytorowej, która obsługiwałaby kilka otoczeń aplikacyjnych jednocześnie. Przykładem tego typu rozwiązania może być np. uruchamianie w jednym otoczeniu aplikacyjnym konwersji zbioru tekstowego utworzonego w jednym standardzie liter polskich na inny standard, w drugim otoczeniu — weryfikacji ortograficznej drugiego tekstu, a w trzecim — normalnej edycji innego tekstu, przy czym wszystkie otoczenia aplikacyjne współpracowałyby z jedną maszyną edytorową.

5.2. Wady rozwiązania

Wadą opisanej koncepcji, przynajmniej w jej opisanej wyżej postaci, jest pewna uciążliwość w przetwarzaniu dużych akapitów. Zmiana akapitów wymaga przeładowania zawartości pola bieżącego, co powoduje, że przy przechodzeniu między dużymi akapitami może nastąpić pewne spowolnienie pracy edytora. Podobna trudność występuje przy posuwaniu się ku początkowi dużego akapitu — czyli przy zmniejszaniu wartości bieżącego przesunięcia — formatowany musi być za każdym razem cały akapit. Trzeba bowiem uzyskać wartość przesunięcia, w którym zaczyna się poprzednia linia tekstu (tzw. formatowanie wsteczne). Rozwiązaniem może być przechowywanie dla każdego akapitu wartości przesunięć względem jego początku kolejnych wielokrotności pewnej stałej liczby linii, np. wartości przesunięć linii 50, 100, 150, 200, 250 itd. Formatowanie wsteczne byłoby realizowane wtedy w porcjach nie większych niż 50 linii. Koncepcja taka wymagałaby odpowiedniego rozszerzenia struktury opisującej akapit PAR.

Trudności z dużymi akapitami są typowe dla edytorów tekstu (np. edytory WORD i WORDSTAR wyraźnie wolniej przesuwają na ekranie duże akapity niż krótsze).

Wady te można zminimalizować lub wręcz usunąć przez wprowadzenie kilku buforów dla akapitu bieżącego (utrzymywać jednocześnie kilka pól bieżących, z których jedno byłoby aktywne). Wymagałoby to dodatkowego sparametryzowania funkcji elementarnych maszyny edytorowej.

5.3. Implementacja

W Instytucie Maszyn Matematycznych zaimplementowano maszynę edytorową, tworząc prosty edytor tekstowy. Użyto kompilatora MICROSOFT C 7.0. Maszyna edytorowa została zaimplementowana jako moduł biblioteczny. W systemie operacyjnym DOS 5.0 moduł ten był dołączany na etapie konsolidacji edytora. W systemie OS/2 (wersja 2.0) przechowany był w bibliotece dynamicznej. Tekst był zorganizowany w dwojaki sposób:

- w łańcuch ze wskaźnikami na następny i poprzedni element w łańcuchu, tak jak zdefiniowano wyżej,

- w tzw. model bąbelkowy, w którym w pamięci operacyjnej przechowywane są tylko akapity modyfikowane (akapity niezmodyfikowane pobierane są z pliku).

Oczywiście sama implementacja maszyny edytorowej nie jest równoznaczna ze stworzeniem pełnego edytora tekstów. Wymaga on utworzenia otoczenia aplikacyjnego (funkcje obsługujące menu, formatujące tekst, realizujące operacje na blokach wierszowych i kolumnowych, przeszukujące tekst i drukujące go itp.). Dopiero te opcje umożliwiają pełną edycję tekstu i czynią edytor atrakcyjnym dla użytkownika. Tym niemniej zastosowanie koncepcji maszyny edytorowej umożliwia łatwą i przejrzystą implementację tych opcji, a także przyszły rozwój edytora i jego przenoszenie do różnych środowisk systemowych.

Literatura

[1] B. W. Kernighan, D. M. Ritchie, Język C, WNT, Warszawa, 1988.

[2] M. J. Rochkind, Advanced C Programming for Displays, Prentice Hall, 1988.

Romuald Synak
Instytut Maszyn Matematycznych WARSZAWA

Laserowe metody uzyskiwania obrazów przestrzennych w przemysłowych systemach widzenia maszynowego

Laser Methods of Range Images Acquisition for Industrial Machine Vision Systems

Streszczenie

Laserowe metody pomiaru odległości pozwalają na rozwiązanie wielu problemów występujących w metrologii, kontroli produkcji i robotyce. Artykuł opisuje metody triangulacyjne i metody polegające na pomiarze czasu przejścia światła oraz przedstawia ich zastosowanie w systemach widzenia maszynowego.

Abstract

Laser range sensing techniques offer a practical solution to many metrological, inspection and robotic problems. This paper describes basic principles of triangulation and time-of-flight techniques and gives a characterization of machine vision industrial applications. The suitability of laser methods for such purposes is considered.

1. Wstęp

Rozwój systemów widzenia maszynowego jest w dużej mierze stymulowany potrzebami, które istnieją w dziedzinie robotyki, metrologii i kontroli produkcji. Zastosowania te stwarzają bowiem wysokie wymagania wynikające z konieczności uzyskiwania obrazów trójwymiarowych i ich przetwarzania w czasie współbieżnym z procesami przemysłowymi. Wyniki pomiarów i przetwarzania muszą odznaczać się przy tym dużą dokładnością.

Złożoność zadań i uwarunkowań ich realizacji spowodowała, że osiągnięcie korzystnych rezultatów jest obecnie możliwe tylko w określonych przypadkach i przy odpowiednim dostosowaniu metod akwizycji obrazów i ich przetwarzania do danego zadania. Metody te tradycyjnie dzieli się na bierne i czynne. Pierwsze dotyczą badania otoczenia tylko przy oświetleniu naturalnym, drugie polegają na wykorzystaniu celowego oświetlenia sceny. Do metod czynnych zalicza się też obszerna grupa oparta na wykorzystaniu ultradźwięków, promieniowania podczerwonego i innych.

Obrazy uzyskiwane za pomocą sensorów, dostosowanych do rodzaju fali przenoszącej informację, można podzielić na intensywnościowe (odtwarzają one rozkład natężenia określonego promieniowania w badanej przestrzeni) i przestrzenne

(zasięgowe — ang. *range image*), wskazujące odległość poszczególnych punktów powierzchni obiektów od płaszczyzny odniesienia [10]. Obrazy intensywnościowe najczęściej uzyskuje się za pomocą kamer CCD i stanowią one wtedy projekcję rozkładu luminancji w przestrzeni na płaszczyznę. Uzyskanie trzeciego wymiaru geometrycznego jest w tym przypadku możliwe na drodze przetwarzania obrazu, przy czym wykorzystuje się wtedy takie jego cechy jak teksturę, zaciemnienie, kontury, kształt, przesunięcie [18]. Opracowano też wiele algorytmów określania odległości na zasadzie stereoskopowej przy wykorzystaniu dwóch kamer [20]. Wadą wymienionych metod jest duża złożoność obliczeniowa i występowanie czynników uniemożliwiających w pewnych przypadkach uzyskanie wyniku, np. znikanie punktów odniesienia przy metodzie stereoskopowej. Z powodu dużego czasu przetwarzania nie jest przy tym na ogół możliwe otrzymanie obrazu w czasie rzeczywistym.

Z powyższych względów do zastosowań przemysłowych najbardziej nadają się metody aktywne, a spośród nich optyczne [19]. Stanowią one szeroką klasę technik, w których stosuje się zarówno źródła światła monochromatycznego (lasery), jak i białego, a pomiaru parametrów geometrycznych dokonuje się przy wykorzystaniu różnych właściwości światła i zjawisk z nim związanych — prędkości rozchodzenia, interferencji, dyfrakcji i innych.

Dla celów uzyskiwania obrazów przestrzennych szczególne znaczenie mają metody laserowe, gdyż pozwalają one określać z dużą dokładnością nie tylko odległość, ale również pozostałe dwie współrzędne punktów obrazu. Wśród tych metod można wyróżnić triangulacyjne oraz oparte na pomiarze czasu przejścia światła [7]. Pierwsze polegają na projekcji wiązki światła laserowego na obiekt i pomiarze przesunięcia obrazu punktu uzyskanego za pomocą kamery. Jeśli chodzi o drugie, to pomiaru czasu przejścia można dokonać mierząc czas propagacji impulsu światła (metoda impulsowa) lub zmianę fazy przebiegu sinusoidalnego modulującego sygnał optyczny (metoda fazowa).

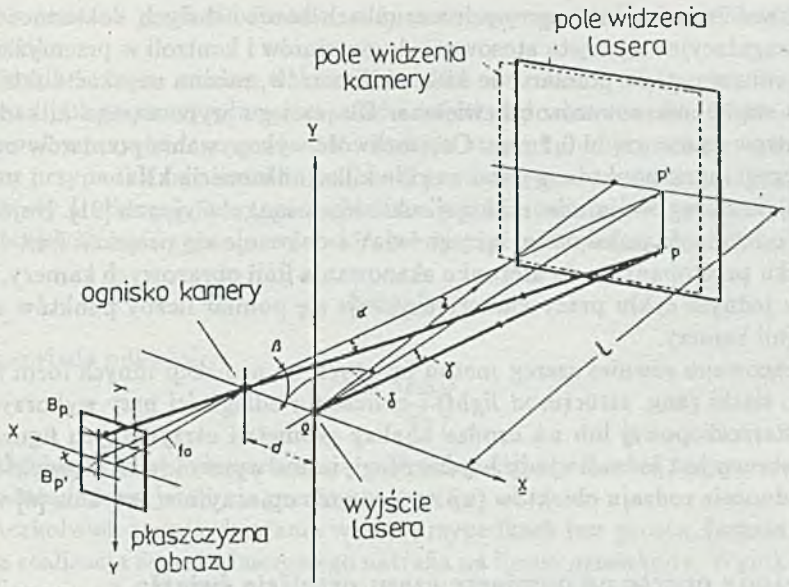
Celem artykułu jest przeanalizowanie, w jakim stopniu metody laserowe spełniają wymagania stawiane systemom widzenia maszynowego przeznaczonym do zastosowań przemysłowych. Analizy takiej dokonano po omówieniu zasadniczych właściwości poszczególnych metod i rozpatrzeniu głównych dziedzin wykorzystania systemów w pomiarach, kontroli i robotyce. Czytelników zainteresowanych innymi metodami optycznymi (np. interferencyjnymi, mory), wykraczającymi poza przyjęty zakres artykułu, kierujemy do literatury [19], [2].

2. Metody akwizycji obrazów za pomocą techniki laserowej

2.1. Metody triangulacyjne

W laserowej metodzie triangulacyjnej wykorzystuje się zarówno kamerę obrazową (lub linijkę fotoelementów), jak i źródło promieniowania. Kamera rejestruje plamkę światła rozproszonego na obiekcie, a położenie punktu odwzorowującego plamkę w płaszczyźnie obrazowej zależy od odległości mierzonego punktu obiektu. Wiązka światła laserowego może być skanowana (odchylana) w kierunku poziomym i pionowym obejmując w ten sposób określony obszar w przestrzeni (Rys. 1).

Jeżeli kamera i źródło światła są oddalone od siebie o odległość d, to szukaną



Rys. 1. Zasada pomiaru triangulacyjnego z użyciem lasera [15]

odległość punktu obiektu l można obliczyć ze wzoru

$$l = \frac{d}{x/f_0 - \mathrm{tg}\delta} \tag{1}$$

gdzie x oznacza współrzędną punktu odwzorującego, δ kąt odchylenia wiązki laserowej, f_0 ogniskową kamery. Rozdzielczość pomiaru odległości Δl jest funkcją minimalnej rozróżnialnej wartości położenia punktu na obrazie (x_m) i kąta odchylenia (δ_m). Definiując rozdzielczość jako

$$\Delta l = l(x + x_m,\ \delta + \delta_m) - l(x, \delta) \tag{2}$$

i dokonując liniowej aproksymacji odpowiedniego wyrażenia można otrzymać

$$\Delta l = \frac{(l^2/d)(-x_m)}{-x_m/f_0 + f_0\delta_m/\sin^2\delta}. \tag{3}$$

Jak stąd wynika, dużą rozdzielczość uzyskuje się dla małych głębokości sceny i dużego odstępu między kamerą i laserem, a także przy dużych kątach odchylenia. Na dokładność pomiaru wpływa też szereg innych czynników, jak zogniskowanie wiązki, zmiana współczynnika odbicia powierzchni [12] i nieciągłości powierzchni. Ponadto występują problemy związane z zasłanianiem punktów, a także występowaniem odbić światła wprowadzających dane nieprawidłowe. Otrzymanie obrazu przestrzennego wymaga przeprowadzenia dużej liczby obliczeń, co wiąże się z potrzebą budowania złożonych procesorów. Mimo wymienionych wad, ze względu

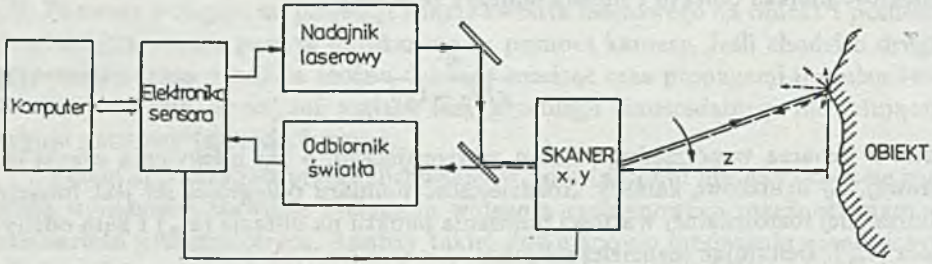
na możliwość uzyskania w pewnych warunkach bardzo dużych dokładności, metody triangulacyjne są często stosowane do pomiarów i kontroli w przemyśle. Przy ograniczeniu np. głębi pomiaru do kilku milimetrów, można uzyskać dokładności pomiaru rzędu mikrometrów lub większe. Dla zasięgu wynoszącego kilkadziesiąt centymetrów są one rzędu 0,1 mm. Częstotliwość wykonywania pomiarów odległości poszczególnych punktów wynosi zwykle kilka–kilkanaście kHz.

Istnieje szereg wariantów realizacji układów triangulacyjnych [21]. Najczęściej zamiast odchylania pojedynczej wiązki światła dokonuje się projekcji linii światła o kierunku prostopadłym do kierunku skanowania linii obrazowych kamery, dzięki czemu w jednym cyklu pracy kamery uzyskuje się pomiar liczby punktów równej liczbie linii kamery.

Opracowano również szereg metod opartych na projekcji innych form świetlnych np. siatki (ang. *structured light*) i obliczaniu odległości przy wykorzystaniu zasady stereoskopowej lub na drodze analizy geometrii otrzymanych figur. Czas analizy obrazu jest jednak wtedy bardzo długi, mimo wprowadzanych zwykle ograniczeń odnośnie rodzaju obiektów (np. w systemie opisanym w artykule [6] wynosi on 5–10 minut).

2.2. Metody oparte na pomiarze czasu przejścia światła

Metody te pozwalają na bezpośredni pomiar odległości obiektu poprzez pomiar czasu propagacji światła od źródła światła do obiektu i po rozproszeniu na nim do odbiornika (Rys. 2).



Rys. 2. Schemat blokowy sensora laserowego działającego na zasadzie pomiaru czasu przejścia światła

Nadajnik i odbiornik światła znajdują się obok siebie i dzięki odpowiedniej konstrukcji układu optycznego odbierana jest składowa sygnału rozproszonego biegnąca współosiowo z wiązką nadawaną. Dzięki temu nie ma zaniku sygnału powodowanego zasłanianiem, co występuje w innych metodach. Urządzenie jest wyposażone w skaner x,y, tj. układ odchylający wiązkę światła w kierunku poziomym i pionowym. Dla każdego wyróżnionego jej położenia dokonuje się pomiaru czasu przejścia światła i po jego zamianie na wymiar odległości dostaje się obraz przestrzenny odwzorowujący odległość w funkcji współrzędnych biegunowych lub prostokątnych.

Odległość obiektu od urządzenia laserowego wynosi

$$l = \frac{ct_p}{2} \tag{4}$$

gdzie c oznacza prędkość światła, a t_p czas propagacji swiatła od nadajnika do odbiornika.

Pomiaru czasu można dokonać dwiema metodami: fazową lub impulsową. Pierwsza polega na wysyłaniu impulsów światła i pomiarze czasu t_p jako opóźnienia impulsu przychodzącego do odbiornika w stosunku do impulsu wysyłanego. W drugim przypadku fala światła jest modulowana sygnałem sinusoidalnym. Detektor fazowy porównuje fazę tego sygnału na wyjściu nadajnika i wejściu odbiornika. Jeżeli częstotliwość sygnału wynosi f, to wystąpi zmiana fazy $\Delta\varphi$ równa

$$\Delta\varphi = 2\pi f t_p \tag{5}$$

co odpowiada odległości

$$l = \frac{c\Delta\varphi}{4\pi f}. \tag{6}$$

Odległość jest więc liniową funkcją mierzonej różnicy fazowej, przy czym wynik jest jednoznaczny, jeżeli nie przekracza ona 2π.

Aczkolwiek zasada działania w obu przypadkach jest prosta, to jednak praktyczna realizacja sensora laserowego natrafia na liczne przeszkody. Wynikają one z jednej strony z krótkich czasów przejścia światła przy odległościach występujących w systemach zorientowanych na zastosowania przemysłowe, a z drugiej strony z możliwościami detekcji odbieranych sygnałów optycznych. Odległość 1 m światło pokonuje w ciągu 3,3 ns, co oznacza, że jeżeli wymagana rozdzielczość wynosi np. 1 mm, czas musi być mierzony z dokładnością większą niż 5 ps. Porównywalne lub większe zmiany opóźnienia mogą wystąpić w samym torze emisji lub wzmocnienia sygnału odbieranego. Konieczne jest zatem stosowanie specjalnych układów z kompensacją tego rodzaju efektów.

Kwestia detekcji sygnału jest związana z poziomem mocy sygnału optycznego dochodzącego do odbiornika. Zależy ona od właściwości rozpraszających obiektu i kwadratu odległości [13]. Światło padając na obiekt ulega częściowo odbiciu i częściowo rozproszeniu, przy czym do odbiornika wraca tylko składowa sygnału rozproszonego, której wielkość zależy od współczynnika rozpraszania obiektu i kątowej charakterystyki rozpraszania. Dla obiektów spełniających np. prawo Lamberta jest ona proporcjonalna do kosinusa kąta zawartego między kierunkiem rozchodzenia się światła i normalną do powierzchni odbijającej. Przy zwykle występujących rozrzutach wszystkich wymienionych wielkości różnice w poziomie sygnału odbieranego mogą sięgać 100 dB. Wzmacniacz odbierający sygnał musi mieć tak dużą dynamikę, by z jednej strony odebrać bardzo niskie sygnały, a z drugiej strony nie wyjść poza zakres liniowy, gdyż może to wpłynąć na wartość mierzonego czasu. Możliwość odbioru sygnałów niskich jest ograniczona z kolei szumami detektora i toru wzmacniającego oraz występującymi zakłóceniami zewnętrznymi.

Metoda fazowa pozwala, jak wynika z literatury [2], na uzyskiwanie większych dokładności pomiaru niż metoda impulsowa. Z kolei wadą metody fazowej jest niejednoznaczność wyników pomiaru powstająca wskutek tego, że tę samą różnicę fazową otrzymuje się dla przesunięć większych o wielokrotność kąta pełnego. Jeżeli zakres zmian odległości jest więc duży, konieczne jest zmniejszenie częstotliwości sygnału modulującego, co prowadzi do pogorszenia dokładności, lub też zastosowanie modulacji kilkuczęstotliwościowej i pewnego skomplikowania budowy sensora.

W praktyce stosowane są obie metody. Na ogół zakres mierzonych odległości jest większy niż kilkanaście–kilkadziesiąt centymetrów i mniejszy niż kilkanaście metrów. Zakres zmian kątów odchylania wynosi kilkadziesiąt stopni. Czas akwizycji obrazu jest zwykle nie większy niż kilka sekund. Parametry przykładowych rozwiązań podano w p. 4.

3. Charakterystyka przemysłowych zastosowań systemów widzenia maszynowego

Zadania, które już obecnie mogą realizować systemy widzenia maszynowego są bardzo różnorodne. Wiąże się z tym duża liczba metod akwizycji obrazów, bardzo różne wymagania co do dokładności zobrazowania oraz stosowanie wielu technik identyfikacji obiektów i ich cech. Dlatego syntetyczne omówienie tej tematyki możliwe jest jedynie w odniesieniu do sytuacji typowych.

3.1. Zastosowanie systemów widzenia maszynowego do pomiarów i kontroli

Użycie systemu wizyjnego pozwala na jednoczesne wykonanie pomiarów parametrów całego obiektu, dzięki czemu skraca się zarówno czas uzyskiwania danych pomiarowych, jak i ich obróbki. Wykorzystanie światła ma przy tym dodatkowe zalety — dużą precyzję pomiaru i jego bezdotykowy charakter. W dalszej części omówimy najczęstsze przypadki zastosowania metod polegających na przetwarzaniu obrazów przestrzennych stanowiących odwzorowanie obiektu w postaci mapy odległości punktów jego powierzchni od płaszczyzny odniesienia [4], [14].

1. Pomiary mikrostruktury powierzchni.

Celem pomiarów jest określenie jakości powierzchni w wyniku obróbki mechanicznej materiału lub innego procesu technologicznego. Przeprowadza się je przy kontroli jakości obróbki polerskiej elementów optycznych lub metalowych (np. łopatek turbiny), oceny połączeń lutowniczych itp. Wynikiem przetwarzania uzyskanego obrazu mogą być parametry określające chropowatość powierzchni, profile wysokości, opis występujących wad i inne. Wymagana jest duża rozdzielczość pomiarów zarówno w kierunku poprzecznym (x lub y), jak i wysokości (z). W wielu zastosowaniach powinna być ona rzędu dziesiątych części mikrometra przy polu pomiarowym od kilku mm^2 do kilku–kilkunastu cm^2. Czas pomiaru i analizy nie jest na ogół krytyczny i może wynosić do kilku minut. Jednak wskutek dużej liczby punktów pomiarowych wymagana częstotliwość ich pomiaru może być od 10 kHz do 1 MHz.

2. Pomiary parametrów geometrycznych obiektów.

Podobnie jak w poprzednim przypadku tworzony jest obraz przestrzenny obiektu. Głównym celem jest wydobycie jego cech geometrycznych takich, jak na przykład rozmiary poszczególnych powierzchni, kąty między nimi, wymiary otworów, występów itp. Pomiary przeprowadza się w celach kontrolnych, a także np. przy tworzeniu modelu cyfrowego obiektu rzeczywistego dla potrzeb systemu projektowania CAD. Wymagane dokładności są od kilku do kilkuset mikrometrów. Przy dużej liczbie powierzchni i dużych rozmiarach stosuje się maszyny pomiarowe zapewniające szeroki zakres zmian w kierunkach x i y za pomocą przesuwów mechanicznych. Pomiaru trzeciego wymiaru dokonuje sensor. Ponieważ czas przesuwu

jest z reguły dosyć długi, potrzebne jest stosowanie wielu sensorów pracujących równolegle lub skanerów. Częstotliwość pomiarów punktowych może dochodzić do 100 kHz.

3. Kontrola prawidłowości montażu.

Zadaniem systemu jest w tym przypadku sprawdzenie, czy w produkowanym wyrobie zostały zamontowane wszystkie elementy i czy znajdują się one we właściwym położeniu. System powinien również wskazać, czy nie ma w nim elementów niepożądanych. Oprócz samego rozpoznania dokonywana jest kontrola wymiarów i wzajemnego usytuowania elementów, przy czym wymagane dokładności są znacznie mniejsze niż w poprzednich zastosowaniach i zawierają się zazwyczaj w granicach od kilku dziesiątych mm do kilku mm. Kontrola może odbywać się bez przerywania produkcji i całkowity jej czas może wynosić od ułamka sekundy do kilku sekund. Zależnie więc od tego i liczby badanych płaszczyzn częstotliwość pomiaru punktów może zawierać się w granicach 10 kHz–1 MHz.

3.2. Zastosowanie systemów widzenia maszynowego w robotyce

Zadania systemów widzenia maszynowego w robotyce można podzielić na dwie grupy:

- rozpoznanie i ocena otoczenia robota oraz wyniku jego pracy,
- nadzorowanie czynności manipulacyjnych i wykonawczych.

System dokonuje wyodrębnienia obiektów w badanej scenie, określa ich cechy jakościowe i ilościowe. Może wykrywać np. stany kolizyjne lub awaryjne, a w przypadku robotów poruszających się, kieruje ich nawigacją. Do omawianej grupy czynności można też zaliczyć kontrolę niektórych parametrów wykonanego wyrobu lub innego rezultatu działania robota.

Jeśli chodzi o drugą grupę funkcji, to system widzenia wyszukuje potrzebny obiekt oraz nadzoruje wykonywanie wielu dalszych czynności jak np. naprowadzanie manipulatora, wyznaczanie jego trajektorii ruchu, regulowanie prędkości przesuwu, uruchamianie narzędzi, czy wykonywanie różnych operacji technologicznych.

Zależnie od realizowanych funkcji zmieniają się wymagania stawiane systemowi widzenia maszynowego. Przy czynnościach kontrolnych i pomiarowych mogą być one podobne do tych, które opisano w p. 3.1. Czynności manipulacyjne wymagają rozpoznania odległości obiektu od manipulatora, przy czym w różnych fazach jego ruchu mogą występować inne wymagania co do dokładności określenia tego parametru. Na ogół wynoszą one od kilku dziesiątych mm do ok. 10 mm przy zasięgu maksymalnym od ok. 10 cm do 1 m. Liczba mierzonych punktów często wynosi 30–100. Wymagany czas cyklu pomiarowego powinien zawierać się w granicach od 0,5 s do 5 s [4]. W wielu przypadkach zachodzi potrzeba śledzenia ruchu manipulatora w sposób ciągły i wówczas wymagania są ostrzejsze — czas ten może wynosić nawet 0,01 s, co przy dużej liczbie punktów prowadzi do częstotliwości pomiaru punktowego do rzędu 1 MHz. Aby uniknąć zasłaniania pola widzenia sensora przez inne elementy, umieszcza się go niekiedy na ramieniu manipulatora (tzw. „*eye-in-hand*").

Przy kierowaniu robotami samojezdnymi zasięg widzenia systemu musi być znacznie większy i może dochodzić do 10 m lub więcej. Pomiar odległości odbywa się przy tym z dokładnością 1% lub większą.

4. Stan rozwoju laserowych metod akwizycji obrazów w aspekcie zastosowań przemysłowych

Prowadzone od kilkunastu lat prace nad wykorzystaniem techniki laserowej do systemów widzenia maszynowego przyniosły szereg znaczących rozwiązań modelowych, a nawet wprowadzenie do produkcji niektórych typów sensorów. Dotyczy to w szczególności sensorów do zastosowań kontrolnych działających na *zasadzie triangulacyjnej*. Poniżej przytoczymy niektóre charakterystyczne dane odnoszące się do modeli sensorów opracowanych dla zastosowań przemysłowych.

Jeśli chodzi o urządzenia pracujące na zasadzie triangulacyjnej, to w grupie systemów przeznaczonych do pomiarów i kontroli można wymienić aparaturę firmy Renishaw Metrology Ltd. [9], obejmującą elementy wykonawcze w postaci sond laserowych oraz układy sterujące i sprzęgające z komputerem. Sondy charakteryzują się rozdzielczością 1 μm przy zakresie pomiarowym 4 mm i częstotliwością pomiarów odległości punktów wynoszącą 100 Hz. Dzięki uniwersalnej konstrukcji mogą być one wykorzystywane do różnych zastosowań, w których zachodzi potrzeba dokonywania pomiarów geometrycznych, np. w maszynach pomiarowych. Większą rozdzielczość pomiaru uzyskuje się w systemach dostosowanych do badania mikrostruktury powierzchni. Przykładem tego może być system, opisany w pracy [3], przeznaczony do kontroli płytek hybrydowych. Urządzenie umożliwia uzyskiwanie obrazów przestrzennych obiektów o wymiarach poprzecznych do kilkunastu milimetrów przy rozdzielczości odległości większej niż 0,375 μm.

Opisane systemy należą do grupy wolnodziałających. Opracowano jednak również systemy odznaczające się dużą prędkością działania i jednocześnie wysoką rozdzielczością. Do nich należy np. urządzenie firmy Synthetic Vision Systems [2] cechujące się częstotliwością uzyskiwania całego obrazu 30 Hz, przy czym obraz zawiera 256 × 256 pikseli o liczbie poziomów odległości 256. Może być ono wykonywane w wersji charakteryzującej się rozdzielczością 1 μm.

Jeśli chodzi o systemy przeznaczone do zastosowań w robotyce, to na zasadzie triangulacyjnej działa seria urządzeń opisanych w pracy [5]. Pierwsze z nich, typu „eye in hand", ma zastosowanie w precyzyjnych manipulatorach i pomiarach. Służy do akwizycji obrazów o zasięgu od 10 cm do 30 cm i rozdzielczości 0,1 mm. Liczba pikseli w obrazie wynosi 512 × 512, a liczba poziomów odległości 256. Częstotliwość pracy jest taka sama, jak dla kamer video. Drugi z systemów jest zaprojektowany na zakres odległości od 0,25 m do 1 m i ma podobną rozdzielczość. Obrazy o liczbie pikseli 256 × 256 uzyskiwane są w ciągu 1 s. Ostatni z systemów ma zakres odległości od 0,25 m do 10 m, rozdzielczość 10 cm i czas uzyskiwania obrazów kilka sekund. Przeznaczony jest do nawigacji pojazdów autonomicznych i wykrywania przeszkód.

Należy również wspomnieć o pracach prowadzonych na uniwersytecie w Oksfordzie nad systemem widzenia maszynowego do prowadzenia pojazdów, w którym odległość mierzy się z dokładnością ok. 1 mm przy zasięgu 5 m [17]. Czas akwizycji obrazu wynosi kilka sekund.

Głównie do zastosowań w robotyce przeznaczone są również systemy działające na *zasadzie pomiaru czasu przejścia światła*.

Jako przykład wykorzystania *metody fazowej* można wskazać urządzenia opracowane przez ERIM (Environmental Research Institute in Michigan) opisane w

pracy [1]. Jedno z nich służy do pomiarów odległości w zakresie od 0,15 m do 0,9 m z rozdzielczością 0,8 mm. Czas akwizycji obrazu wynosi od 2 s do 10 s. Drugie umożliwia pomiary w zakresie do 19,4 m. Rozdzielczość wynosi 8 cm, a częstotliwość pomiaru punktów 92 kHz. Na politechnice w Monachium opracowano urządzenie dokonujące analizy sceny w zakresie do 15 m [8]. Obraz zawiera 48 × 81 pikseli i jest uzyskiwany w ciągu 0,4 s. Dokładność pomiaru zależnie od odległości i rodzaju obiektu wynosi od 7 mm do 7 cm.

Jeśli chodzi o systemy wykorzystujące *metodę impulsową*, to w pracy [11] opisano urządzenie przeznaczone do uzyskiwania obrazów o liczbie pikseli 512 × 512 przy zasięgu do 4 m. Dokładność pomiaru zawiera się w granicach ±7 mm, a czas otrzymania obrazu wynosi 26 s. Znacznie szersze zakresy pomiaru wynoszące od ok. 1 cm do 100 m uzyskano w urządzeniu opisanym w pracy [16]. Przy częstotliwości pomiarów 10 kHz dokładność pomiaru wynosi 9 mm, jednak można ją znacznie poprawić na drodze obróbki wyników uzyskanych przy zwielokrotnieniu liczby impulsów — np. przy 100 impulsach wynosi ona 1 mm.

Porównując wyniki uzyskane w dziedzinie budowy sensorów laserowych z wymaganiami stawianymi przez zastosowania przemysłowe, można stwierdzić, że w wielu przypadkach są one zgodne z oczekiwaniami. Dotyczy to w szczególności mierzenia i kontroli za pomocą metod triangulacyjnych, które pozwalają na osiągnięcie wymaganych dokładności. W niektórych przypadkach za mała może okazać się częstotliwość pomiarów wynikająca często z dużej bezwładności przesuwów mechanicznych. Jeśli chodzi o zastosowania w robotyce, to porównywalne wyniki uzyskuje się za pomocą wszystkich opisanych metod. Są one jednak zadowalające, gdy obiekty badanej sceny charakteryzują się dużą jednorodnością pod względem rozpraszania i odbijania światła. Należy i tu oczekiwać jednak postępu w miarę rozwoju metod bazujących na modelowaniu obiektów, a także w wyniku zwiększania prędkości przetwarzania danych pomiarowych. Wydaje się, że preferowane będą wtedy metody polegające na pomiarze czasu propagacji światła.

5. Zakończenie

Wykorzystanie techniki laserowej do akwizycji obrazów trójwymiarowych pozwala na spełnienie trudnych wymagań, które są stawiane systemom widzenia maszynowego w ich przemysłowych zastosowaniach. Daje ona możliwość dokładniejszego i szybszego w porównaniu z innymi metodami określania odległości punktów w analizowanym obszarze. Dodatkową korzyścią jest ograniczanie wpływu takich czynników jak oświetlenie sceny, występowanie cieni, zasłanianie, znikanie punktów odniesienia i innych, które wiążą się z użyciem metod pasywnych.

Realizacja sensorów laserowych wymaga jednak pokonania szeregu trudności, które wynikają z wpływu wielu czynników na wielkość sygnału optycznego i możliwości jego detekcji. Do nich należą właściwości rozpraszające badanego obiektu, występowanie dodatkowych odbić światła, zakłócenia i szumy w torze odczytu. Komplikuje to budowę sensorów i powoduje, że ich koszt jest wysoki.

Z drugiej strony należy wspomnieć o trwającym w dalszym ciągu rozwoju laserów półprzewodnikowych, które są obecnie podstawowym źródłem światła w sensorach wizyjnych. Niedawno wprowadzono do produkcji lasery emitujące światło czerwone, eksperymentalnie uzyskuje się już lasery o innych długościach fali

z zakresu widzialnego, następuje zwiększenie mocy promieniowania i poprawa parametrów optycznych wiązki światła. Stwarza to nowe możliwości zarówno w rozpoznawaniu obiektów, jak i poprawie dokładności pomiarów. Wzrost produkcji laserów i innych elementów optoelektronicznych przyczyni się do obniżenia ceny sensorów. Należy więc oczekiwać, że przy złożonych zadaniach będzie stosowanych kilka sensorów laserowych o wyspecjalizowanych funkcjach.

Literatura

[1] M.E. Bair et al., Three-dimensional imaging and applications, SPIE v. 726, Intell. Robots and Comp. Vision, 1986, 264–274.

[2] L.H. Bieman, Three-dimensional machine vision, Phot. Spectra no. 5, 1990, 81–90.

[3] S.K. Case, Laser sensors for precise control of hybrid manufacturing, Hybrid Circuit Technology, no. 1, 1991, 5–11.

[4] N.R. Corby, J.L. Mundy, Applications of range image sensing and processing, Analysis and Interpretation of Range Images, R.C. Jain, A.K. Jain (ed.), Springer V., 1990, 255–272.

[5] A. Guinet et al., 3D measurement from 10 cm to 10 m, SPIE v. 1010, Industrial Inspection, 1988, 83–90.

[6] G. Hu, G. Stockman, 3-D surface solution using structural light and constraint propagation, IEEE Tr. PAMI, v. 11, no. 4, 1989, 309–402

[7] R.A. Jarvis, A perspective on range finding techniques for computer vision, IEEE Tr. PAMI, v. 5, no. 2, 1983, 122–139.

[8] G. Karl, G. Schmidt, Die Umwelt dreidimensional erfassen, Elektronik, no. 19, 1989, 78–88.

[9] A. Modjarrad, Non-contact measurement using a laser scanning probe, SPIE v. 1012, Industrial Inspection, 1988, 83–90.

[10] W.Mokrzycki, Wprowadzenie do maszynowego postrzegania, Informatyka, nr 10, 1990, 1–8.

[11] I. Moring et al., Acquisition of three-dimensional image data by a scanning laser range finder, Opt. Eng., 8, 1989, 897.

[12] J.L. Mundy, G.B. Porter, A three-dimensional sensor based on structured light, Three-Dimensional Machine Vision, T. Kanade (ed.), Kluver A.P. 1987, 3–61.

[13] D. Nitzan et al., The measurement and use of registered reflectance and range data in scene analysis, Pr. IEEE, v. 65, no. 2, 1977, 206–220.

[14] T.P. Pryor, Applications of lasers to production, metrology control and machine vision, Pr. IEEE, v. 70, no. 6, 1982, 618–625.

[15] U. Rembold et al., The role of the computer in robot intelligence, Handbook of Industrial Robotics, S. Y. Not (ed.), J. Wiley, 1985, 437–463.

[16] R. Schwarte, Multiresolutional laser radar, Traditional and Non-Traditional Robotic Sensors, T.C. Henderson (ed.), Springer V., 1990, 127–142.

[17] –, Smart sensor guide automated vehicles, Phot. Spectra, no. 1, 1991, 36.

[18] Y. Shirai, Three-dimensional Computer Vision, Springer V., 1987.

[19] T.C. Strand, Optical three-dimensional sensing for machine vision, Opt. Eng., v. 24, no. 1, 1985, 33–40.

[20] S. Tanda, A.C. Kak, A rule-based approach to binocular stereopsis, Analysis and Interpretations of Range Images, R.C. Jain, A.K. Jain (ed.), Springer V., 1990. 33–139.

[21] P. Vuylsteke et al., Three dimensional image acquisition, Traditional and Non-Traditional Robotic Sensors, T. C. Henderson (ed.), Springer V., 1990, 187–210.

JAROSLAW WÓJTOWICZ
INSTYTUT MASZYN MATEMATYCZNYCH WARSZAWA

Wirtualny system pomiarowy VIRT II
Virtual Measurement System VIRT II

W Instytucie Maszyn Matematycznych został zaprojektowany i uruchomiony wirtualny system pomiarowy VIRT II przeznaczony do pomiarów wielkości elektrycznych. System składa się z kart pomiarowych instalowanych w komputerach klasy PC i oprogramowania użytkowego. Karty pomiarowe należą do najtańszych wirtualnych przyrządów pomiarowych, to jest takich, w których opropgramowanie zainstalowane na zewnątrz przyrządu (w komputerze) decyduje o możliwości wykorzystania przyrządu.

W skład systemu wchodzą następujące karty:

RC681A mostek RLC, umożliwia określenie 17 parametrów immitancyjnych dwójników z dokładnością 0,5% i szybkością 2 pomiarów na sekundę;

AD611B wielokanałowy przetwornik A/C i C/A, 16 kanałów wejściowych, 1 kanał wyjściowy, wyjście napięcia odniesienia, rozdzielczość 12 bitów, dokładność podstawowa 0,05%, czas konwersji 15 μs;

IO601A moduł cyfrowych wejść/wyjść, wejście licznika 8253 oraz 48 do 144 linii we/wy obsługiwanych przez układy 8255;

PG621A generator przebiegów arbitralnych, generuje przebieg o dowolnym kształcie z rozdzielczością 8 bitów i szybkością ustawiania próbek 2 MHz, jeden okres przebiegu można zbudować z 32000 próbek;

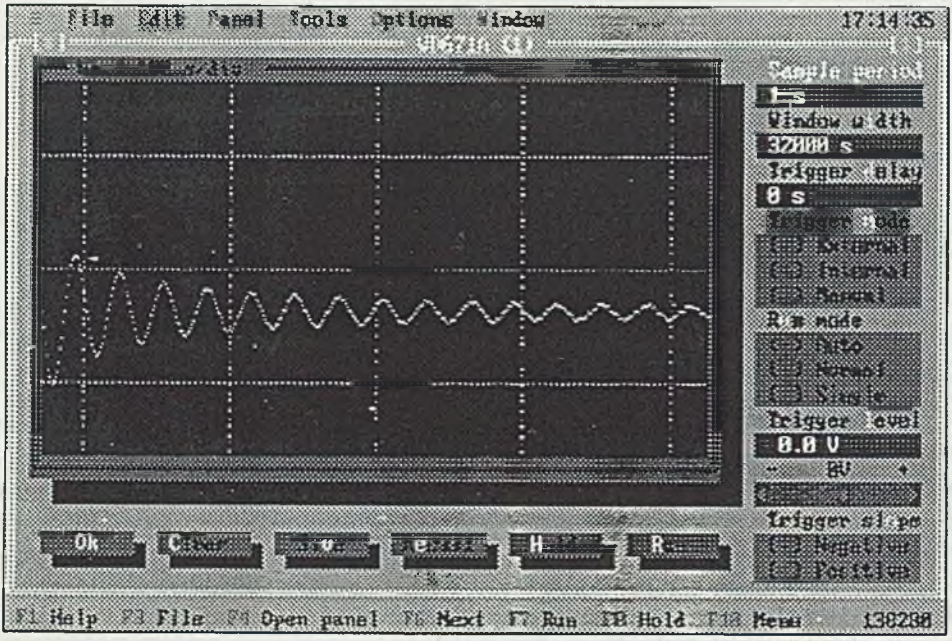
WD671A rejestrator przebiegów jednorazowych, rozdzielczość 8 bitów, częstotliwość próbkowania 10 MHz, rejestracja 64536 próbek.

Każda karta posiada własną bibliotekę procedur obsługi napisaną w Turbo C 2.0 i Turbo Pascalu 5.0.

Cechą systemu VIRT II jest możliwość dowolnego skonfigurowania systemu pomiarowego. Użytkownik definiuje za pomocą programu użytkowego ilość kart pomiarowych dołączonych do systemu, określa ich adresy w przestrzeni I/O komputera i ustala parametry pomiarów. W czasie konfigurowania systemu każdemu kanałowi pomiarowemu zostanie automatycznie przypisane okno, w którym będą wyświetlane cyfrowo (lub graficznie — dla karty WD671A) wyniki pomiarów.

Oprogramowanie systemowe pozwala za pomocą myszy lub kursora wskazać, które z okien mają być wyświetlane na ekranie monitora, zaprojektować wielkość ich położenia na ekranie, zmieniać zakresy pomiarowe, umieszczać komentarze i obliczenia pośrednie. Do systemu jest dołączony także podręczny kalkulator.

Na Rys. 1 przedstawiono przykładowo obraz na monitorze dla karty WD671A.



Rys. 1. Płyta czołowa WD671A z zarejestrowanym sygnałem

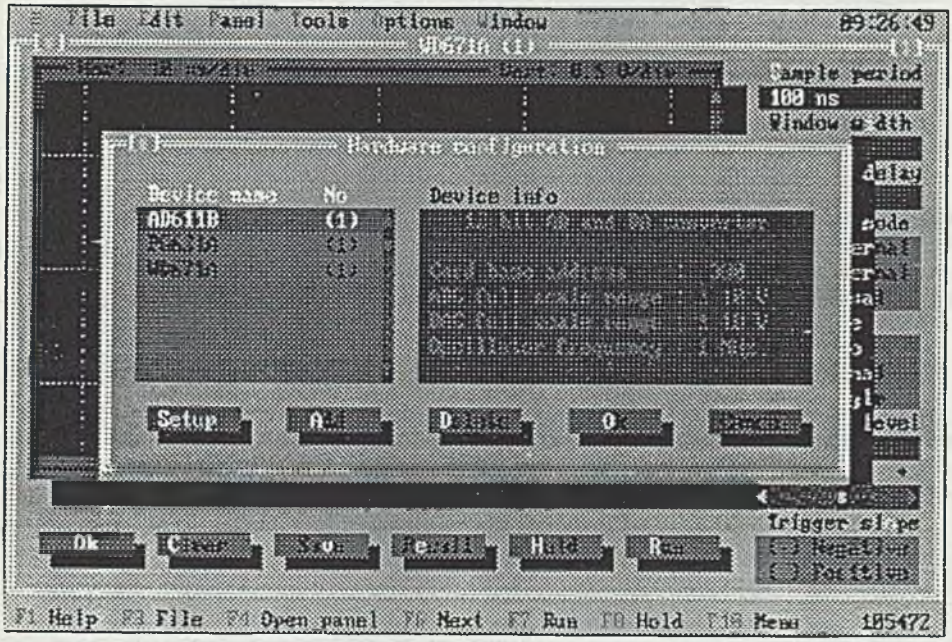
Główną część ekranu zajmuje pole wyświetlania w formie graficznej zarejestrowanego sygnału, z prawej strony są opisane parametry rejestracji przebiegu, a u dołu „klawisze" zerowania pola, zapisu na dysk, zatrzymania oraz rozpoczęcia rejestracji.

Dla potrzeb VIRT-a II została także opracowana zdalnie sterowana autonomiczna jednostka systemu pomiarowego, składająca się z obudowy i zasilacza, płyty głównej PC 286, karty I/O z RS-232, kart pomiarowych i (EP)ROM-DISK-u.

(EP)ROM-DISK jest kartą instalowaną w komputerze PC i całkowicie zastępującą dowolny mechanizm dysków miękkich. Program dołączony do karty pozwala użytkownikowi na korzystanie z (EP)ROM-DISK-u tak jak z dyskietki zabezpieczonej przed zapisem. Po włączeniu komputera z (EP)ROM-DISK-u jest ładowany system operacyjny i uruchamiany dowolny program sterujący systemem. Daje to możliwość całkowicie automatycznego działania systemu pomiarowego, albo przy użyciu odpowiednich procedur komunikacyjnych jednostka pomiarowa może być sterowana z komputera głównego przez port szeregowy RS-232. (EP)ROM-DISK można stosować w każdym innym rozwiązaniu wymagającym zastąpienia mechanizmu dyskowego.

Trwają prace nad rozwinięciem systemu, a w szczególności nad stworzeniem systemu kontrolno-pomiarowego z lepszym wykorzystaniem cech wirtualnych, takich jak możliwość określenia przez użytkownika struktury systemu pomiarowego i sekwencji pomiarów. Powstanie również nowy, znacznie wygodniejszy graficzny interfejs użytkownika. System zostanie uzupełniony o możliwość tworzenia własnych

poleceń (makropoleceń) zbudowanych z podstawowych rozkazów zawartych w bibliotekach procedur każdej z kart. Użytkownik określi w jakiej kolejności i z jaką częstotliwością mają być wykonane pomiary, jakie karty mają być w cyklu pomiarowym wykorzystane i gdzie zebrane dane należy rejestrować: do zbioru na dysku, do pamięci, lub tylko wyświetlić na monitorze. Każde takie makropolecenie można zapamiętać i ewntualnie wykorzystać podczas innych pomiarów. Użytkownik może stworzyć bibliotekę makropoleceń służących jego indywidualnym potrzebom. Wyeliminowana zostanie konieczność pisania i kompilowania programów sterujących systemem pomiarowym. Dla ułatwienia w projektowaniu makropoleceń jest przewidziany tryb pracy ręcznej (krokowy).



Rys. 2. Konfigurowanie systemu pomiarowego

ANDRZEJ KACZMARCZYK
INSTYTUT MASZYN MATEMATYCZNYCH WARSZAWA

Systemy wytwórcze do edukacji w komputerowo zintegrowanym wytwarzaniu (CIM)

Manufacturing Systems for Education in CIM

Streszczenie

Nowe środowisko technologiczne powstające w przemyśle w wyniku wprowadzania technik CIM stwarza zapotrzebowanie na specjalistów CIM. Edukacja takich specjalistów wymaga odpowiednich systemów szkoleniowych. W artykule przedstawiono dotychczasowy rozwój i stan obecny szkoleniowych systemów wytwórczych, perspektywy dalszego rozwoju i wynikające stąd wnioski.

Abstract

New manufacturing environment, created in industry by CIM technologies, implies demand for CIM profesionalists. CIM education requires convenient training systems. Present state of such systems development, as well as perspectives of that development, are discussed in the paper, and suggestions are formulated.

1. Wstęp

Ogólnym celem, któremu służą wysiłki finansowe, organizacyjne i techniczne w przemyśle, jest utrzymanie zdolności konkurencyjnej przedsiębiorstwa — w skali mikro — i całego krajowego przemysłu w skali makro. Konkurencyjność, *competitiveness*, jest hasłem końca stulecia*.

Aby utrzymać zdolność konkurencyjną przedsiębiorstwa, trzeba ustawicznie zastępować stare wyroby nowymi, oferować je w dużej liczbie odmian, zapewnić wysoką jakość, krótkie terminy dostaw i niskie ceny. Spełnienie tych wymagań jest dziś możliwe jedynie przy rozszerzającej się komputerowej automatyzacji i/lub komputerowym wspomaganiu wszelkich rodzajów działalności w przedsiębiorstwie oraz komputerowej integracji tej działalności. W szybkim tempie powstaje więc w krajach rozwiniętych nowe środowisko technologiczne, na które składają się wysokowydajne procesy obróbcze, roboty i komputery o zaawansowanej inteligencji, wyposażone w wiedzę ekspercką, połączone lokalnymi sieciami komunikacyjnymi,

* W USA dwie kluczowe ustawy technologiczne to: National Competitiveness Act (H.R. 5231) oraz Manufacturing Strategy Act (S. 1330), które w br. mają być scalone w jedną ustawę.

wplecionymi z kolei w sieci rozległe, stwarzające dostęp do publicznych usług komunikacyjnych i informacyjnych.

Wydatkowano już na to i nadal wydatkuje się duże środki z funduszy państwowych i z funduszy firm prywatnych, m.in. w ramach programów ESPRIT* i EUREKA w Europie, w ramach programów Narodowej Fundacji Wiedzy (NSF) oraz Narodowego Instytutu Norm i Technologii (NIST) w USA, a w Japonii — Ministerstwa Międzynarodowego Handlu i Przemysłu (MITI). Właśnie w Japonii wkracza obecnie w fazę realizacji międzynarodowy program 1990–2000 „Inteligentne Systemy Wytwórcze", z udziałem krajów EWG oraz USA i z budżetem 1,5 miliarda dolarów, którego celem jest opracowanie „systemu produkcyjnego XXI wieku".

Z nowym środowiskiem technologicznym najczęściej kojarzona jest nazwa „Komputerowo zintegrowane wytwarzanie" (*CIM — Computer Integrated Manufacturing*). Komputerowa integracja obejmuje wszystkie funkcje przedsiębiorstwa i wszystkie jego komórki organizacyjne, zapewniając przepływ informacji pomiędzy ludźmi, maszynami technologicznymi i komputerowymi bankami informacji.

Na Rys. 1 pokazano koncepcję CIM w przedsiębiorstwie, przyjętą w niemieckim programie *CIM Einfuehrungen*, finansowanym przez Federalne Ministerstwo Badań i Technologii.

Na Rys. 2 przedstawiono schemat komputerowej sieci komunikacyjnej dla CIM w przedsiębiorstwie według materiałów Międzynarodowej Federacji Użytkowników MAP/TOP, tzn. użytkowników standardów pn. Protokoły Automatyzacji Produkcji (MAP) i Protokoły Techniczne i Biurowe (TOP). Standardy MAP/TOP, zgodne z zasadami określonymi w międzynarodowych normach ISO (*the International Organization for Standardization*) dla tzw. systemów otwartych (OSI), umożliwiają realizację CIM przy użyciu sprzętu różnych producentów.

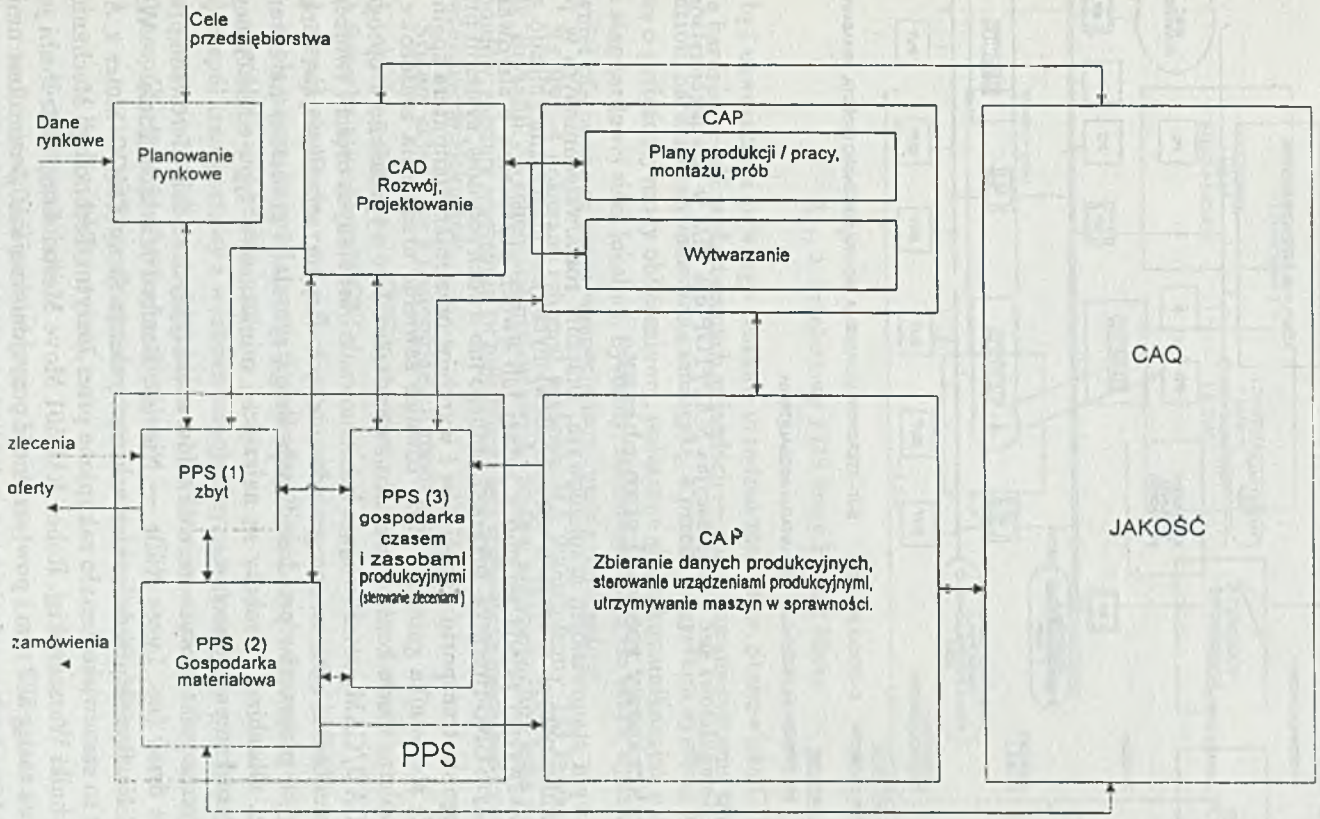
Problemem newralgicznym, wysuwanym na czołowe miejsce w ocenach sytuacji obecnej i rozważaniach na temat przyszłości CIM, jest problem kształcenia kadr — menedżerskich i technicznych, na wszystkich poziomach edukacji — zdolnych do wykorzystania możliwości jakie daje ta nowa technologia i do jej rozwijania. Wszystkie kraje uprzemysłowione odczuwają dziś niedostatek profesjonalistów CIM.

Wyszkolenie kadr dla CIM jest przedsięwzięciem złożonym, w którym przekazywanie wiedzy z zakresu dyscyplin podstawowych i stymulowanie rozwoju ogólnego spełnia rolę znaczącą, jednak nieodzownym składnikiem edukacji musi być nauczenie działania w środowisku technologicznym CIM. Rzeczywiste systemy funkcjonujące w przemyśle są bardzo kosztowne, muszą być intensywnie eksploatowane i dlatego nie nadają się do szkolenia i treningu. Powstają coraz liczniejsze systemy wytwórcze przeznaczone specjalnie do celów edukacyjnych, instalowane w laboratoriach dydaktycznych, wykorzystywane zarówno w edukacji szkolnej, jak i w kształceniu ustawicznym osób zatrudnionych w przemyśle.

Przedmiotem artykułu są systemy wytwórcze dydaktyczne, przeznaczone do edukacji w zakresie CIM. Przedstawiony będzie ich dotychczasowy rozwój i stan obecny oraz wizja dalszego rozwoju i wynikające z niej wnioski.

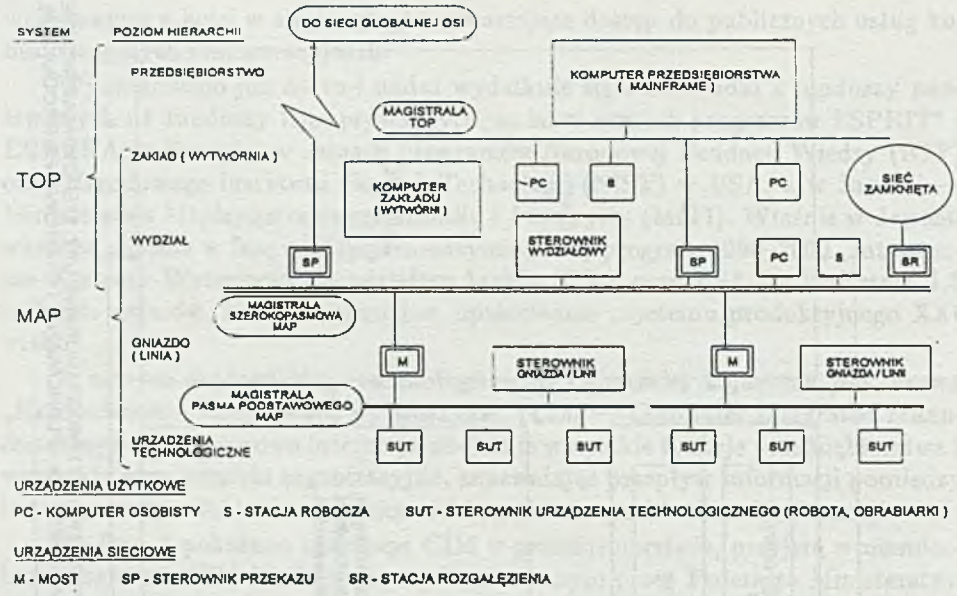
---

* W pięcioleciu 1984–89 13% całej pracochłonnosci ESPRIT wynoszącej 7200 osobo-lat.



CAD - ROZWÓJ I KONSTRUKCJA
CAM - WYKONANIE , TRANSPORT , MONTAŻ
CAP - PLANOWANIE PROD. / PRACY I PROGRAMOWANIE URZ. STEROW. NUMERYCZNIE
CAQ - ZAPEWNIENIE JAKOŚCI PPS - PLANOWANIE I STEROWANIE PRODUKCJĄ

Rys. 1. CIM w przedsiębiorstwie



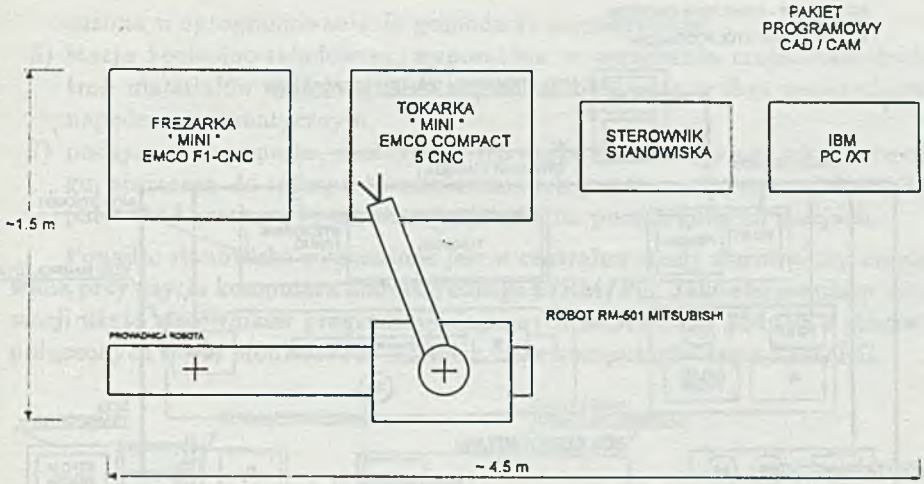
Rys. 2. Komputerowa sieć komunikacyjna MAP/TOP dla CIM w przedsiębiorstwie

2. Dotychczasowy rozwój i stan obecny

Bazowym stanowiskiem szkoleniowym, znacznie rozpowszechnionym w placówkach edukacyjnych w krajach uprzemysłowionych, jest elastyczne gniazdo wytwórcze. Zazwyczaj składa się ono z jednej lub dwóch obrabiarek o sterowaniu numerycznym CNC (frezarka, tokarka), jednego lub dwóch robotów, systemu przenośnikowego do transportu materiałów i wyrobów oraz mikrokomputera spełniającego rolę sterownika gniazda. Może również zawierać urządzenia sensorowe, a także kompletne stacje kontroli kształtu i wymiarów. Jest wyposażone w oprogramowanie CAD/CAM, umożliwiające konstruowanie obrabianych części i tworzenie programów obróbki oraz w oprogramowanie symulacyjne, umożliwiające sprawdzenie owych programów przed realizacją drogą symulacji graficznej na ekranie komputera. Obrabiarki i roboty są najczęściej miniaturowe, typu edukacyjnego. Stosuje się tu komputery osobiste.

Elastyczne gniazda wytwórcze do celów edukacyjnych są dziś oferowane przez dość liczne firmy (np. Lucas Nülle — Niemcy, Denford Machine Tools — Wlk. Brytania). Na Rys. 3 przedstawiono schemat gniazda firmy EMCO Maier z Austrii, które to stanowisko zostało zakupione przez Instytut Technologii Mechanicznej Politechniki Warszawskiej. Robot RM-501 Move Master firmy Mitsubishi jest 5-osiowy, ma zasięg 500 mm i powtarzalność pozycjonowania ±0,5 mm. Jest umieszczony na torze jezdnym, po którym może być przemieszczany do jednego z dwóch możliwych położeń, naprzeciwko każdej z dwóch znajdujących się na stanowisku miniobrabiarek o sterowaniu CNC.

Drugim, bardziej zaawansowanym typem obiektu szkoleniowego, jest system/stanowisko CIM: określone stanowisko laboratoryjne, o wybranej konfiguracji, mo-



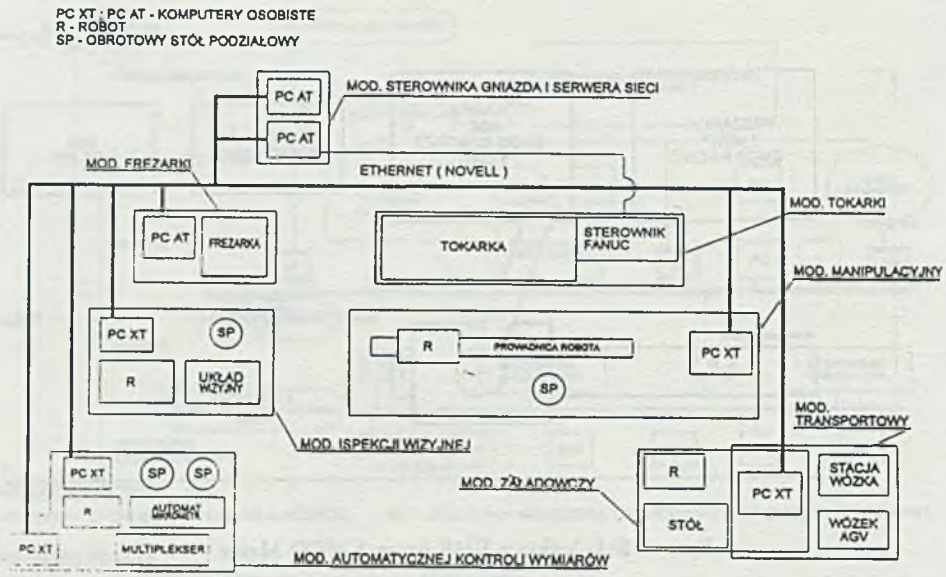
Rys. 3. Szkoleniowy FMS firmy EMCO Maier

że być utworzone z pewnego zestawu urządzeń-modułów. Stanowiska CIM zawierają bardziej — niż w elastycznych gniazdach — rozbudowane systemy transportu, kontroli, bardziej zaawansowaną sensorykę, a przede wszystkim systemy komputerowe o rozłożonej mocy obliczeniowej, połączone siecią komunikacyjną o charakterze standardowej sieci lokalnej. Systemy/stanowiska CIM do celów dydaktycznych również oferowane są przez wyspecjalizowanych producentów.

Na Rys. 4 pokazano schemat systemu MA 9000/5 firmy TQ International z Wlk. Brytanii. Na schemacie przedstawiono zestaw połączonych ze sobą wszystkich modułów; moduły występują tu pojedynczo lecz stanowiska mogą zawierać moduły powtórzone wielokrotnie.

System składa się z następujących modułów:

1) tokarka o sterowaniu CNC z automatyczną wymianą narzędzi;
2) frezarka o sterowaniu CNC;
3) zespół manipulacyjny z 5-osiowym robotem umieszczonym na liniowym zespole przejezdnym dającym dodatkowy stopień swobody, zawierający również obrotowy stół podziałowy;
4) zespół transportowy z wózkiem samojezdnym, wyposażony w trzy palety transportowe z koszami, w tym jedna z urządzeniem samozaładowczym;
5) zespół inspekcji wizyjnej zawierający:
   - system wizyjny służący do rozpoznawania sylwetki obiektu umieszczonego na podświetlonym stole,
   - 6-osiowego robota,
   - obrotowy stół podziałowy;
6) stacja załadowcza (wejściowa), składająca się ze stołu z urządzeniem samozaładowczym oraz 6-osiowego robota;
7) zespół automatycznej kontroli wymiarów zawierający:
   - automatyczny (z własnym napędem) mikrometr,
   - 6-osiowego robota,



Rys. 4. System/stanowisko CIM firmy TQ International

- 2 obrotowe stoły podziałowe;

8) sterownik stanowiska i sieć lokalna Novell z oprogramowaniem ELS Level 2 NetWare;
9) oprogramowanie CAD/CAM oraz symulacyjne, składające się ze zintegrowanego pakietu Auto CAD/PEPS 2 CAD CAM.

Moduły te zawierają łącznie osiem komputerów typu IBM/PC i cztery roboty; stosowane są interfejsy RS 232.

Na Rys. 5 pokazano schemat stanowiska szkoleniowego CIM-2000 firmy DEGEM Systems z Izraela. Stanowisko, zbudowane przy użyciu miniobrabiarek i minirobotów, składa się z następujących stacji-podsystemów:

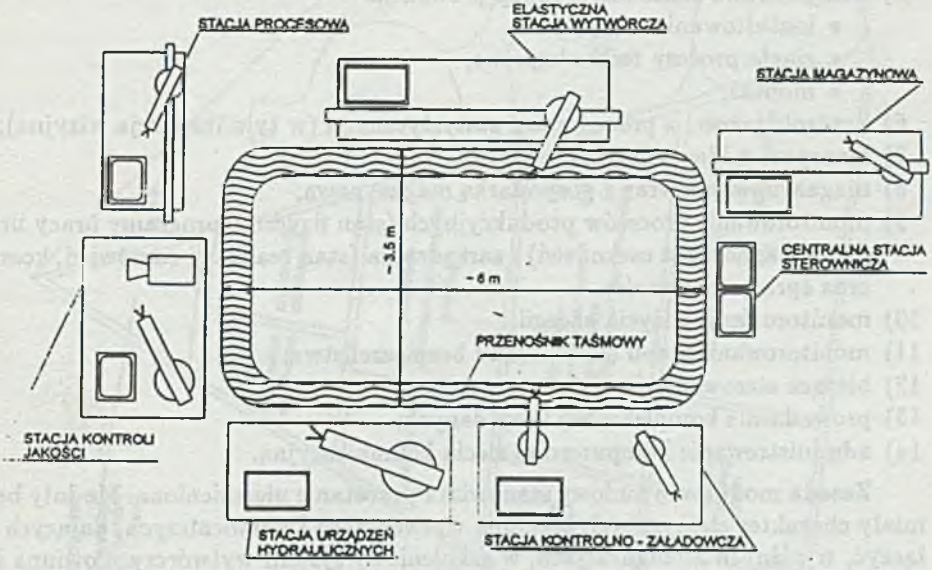
1) elastyczna stacja wytwórcza, odpowiadająca omówionemu poprzednio elastycznemu gniazdu wytwórczemu, złożona z tokarki i frezarki o sterowaniach CNC oraz z 5-osiowego robota o zasięgu 410 mm i powtarzalności pozycjonowania ±0.3 mm, wyposażona w oprogramowanie CAD/CAM i symulacyjne;
2) stacja procesowa, do prowadzenia ciągłego procesu technologicznego obróbki cieplno-chemicznej, zawierająca sześć pojemników z wyposażeniem (grzejniki, pompy, zawory, armatura, czujniki) do prowadzenia procesu, suszarkę oraz 3-osiowego robota;
3) stacja kontroli jakości z układem wizyjnym i robotem (takim jak w stacji 1);
4) stacja urządzeń hydraulicznych o ciśnieniu roboczym 50 bar, składająca się z 4-osiowego robota o napędzie hydraulicznym, miniaturowej prasy hydraulicznej oraz zasilacza hydraulicznego;
5) 32-miejscowa stacja magazynowa z 3-osiowym manipulatorem, do składowania, na miniaturowych paletach, materiałów, półproduktów i wyrobów, wypo-

sażona w oprogramowanie do gospodarki magazynowej;

6) stacja kontrolno-załadowcza, wyposażona w urządzenia czujnikowe do kontroli materiałów wejściowych do dalszej obróbki oraz w dwa manipulatory z napędem pneumatycznym;

7) podsystem transportu, złożony z przenośnika taśmowego o zamkniętym obiegu, służącego do transportu miniaturowych palet, oraz systemu identyfikacji palet (kod kreskowy) i ich zatrzymywania na poszczególnych stacjach.

Ponadto stanowisko wyposażone jest w centralną stację sterowniczą, zrealizowaną przy użyciu komputera kompatybilnego z IBM/PC. Jako sterowników innych stacji użyto sterowników programowo-logicznych MODICON 984 (na 4 stacjach), połączonych siecią ModBusPlus Network. oraz komputerów typu IBM/PC.



Rys. 5. Stanowisko szkoleniowe CIM firmy DEGEM

Oprogramowanie, oprócz bazowego dla każdej stacji, umożliwiającego tworzenie i wykonywanie programów użytkowych, zawiera programy pozwalające na gromadzenie danych produkcyjnych dla całego stanowiska, jego synoptykę komputerową, wyznaczanie marszrut technologicznych i tworzenie programu pracy całego stanowiska oraz koordynację wykonywania tego programu.

3. Dalszy rozwój

Omówione powyżej, oferowane obecnie stanowiska edukacyjne, realizują przede wszystkim ideę CAD/CAM, tzn. komputerowego sprzężenia procedur projektowych z elastycznie zautomatyzowanym wytwarzaniem. Punkt ciężkości oferowanych rozwiązań leży blisko „podłogi warsztatu".

Można postulować dalszy rozwój polegający na rozbudowywaniu funkcji — i służących do ich realizacji podsystemów stanowiska szkoleniowego — które można

przypisać wyższym poziomom struktury fabryki, widzianej jako piramida z centralnym zarządem w wierzchołku i materialnymi procesami wytwarzania u podstawy.

Takie ukierunkowanie rozwoju jest uzasadnione potrzebą szkolenia adeptów CIM w zakresie dostępu i efektywnego wykorzystania informacji gromadzonej i krążącej w całym systemie. CIM stwarza wielkie możliwości w tym względzie, lecz ich uruchomienie i wykorzystanie wymaga wiedzy i treningu.

Funkcje rozwiniętego stanowiska szkoleniowego CIM:

1) przyjmowanie i rejestracja zamówień;
2) konstruowanie wspomagane komputerowo;
3) planowanie zaopatrzeniowe*;
4) planowanie i harmonogramowanie produkcji;
5) komputerowo zautomatyzowana produkcja:
   - kształtowanie części,
   - ciągłe procesy technologiczne,
   - montaż;
6) kontrola jakości z procedurami statystycznymi (w tym inspekcja wizyjna);
7) transport z identyfikacją części;
8) magazynowanie wraz z gospodarką magazynową;
9) monitorowanie procesów produkcyjnych (stan fizyczny, programy pracy urządzeń, diagnostyka uszkodzeń) i zarządzania (stan realizacji zamówień, koszty) oraz sprawozdawczość;
10) monitorowanie zużycia energii;
11) monitorowanie stanu środowiska i bezpieczeństwa;
12) bieżące sterowanie produkcją;
13) prowadzenie kompleksowej bazy danych;
14) administrowanie komputerową siecią komunikacyjną.

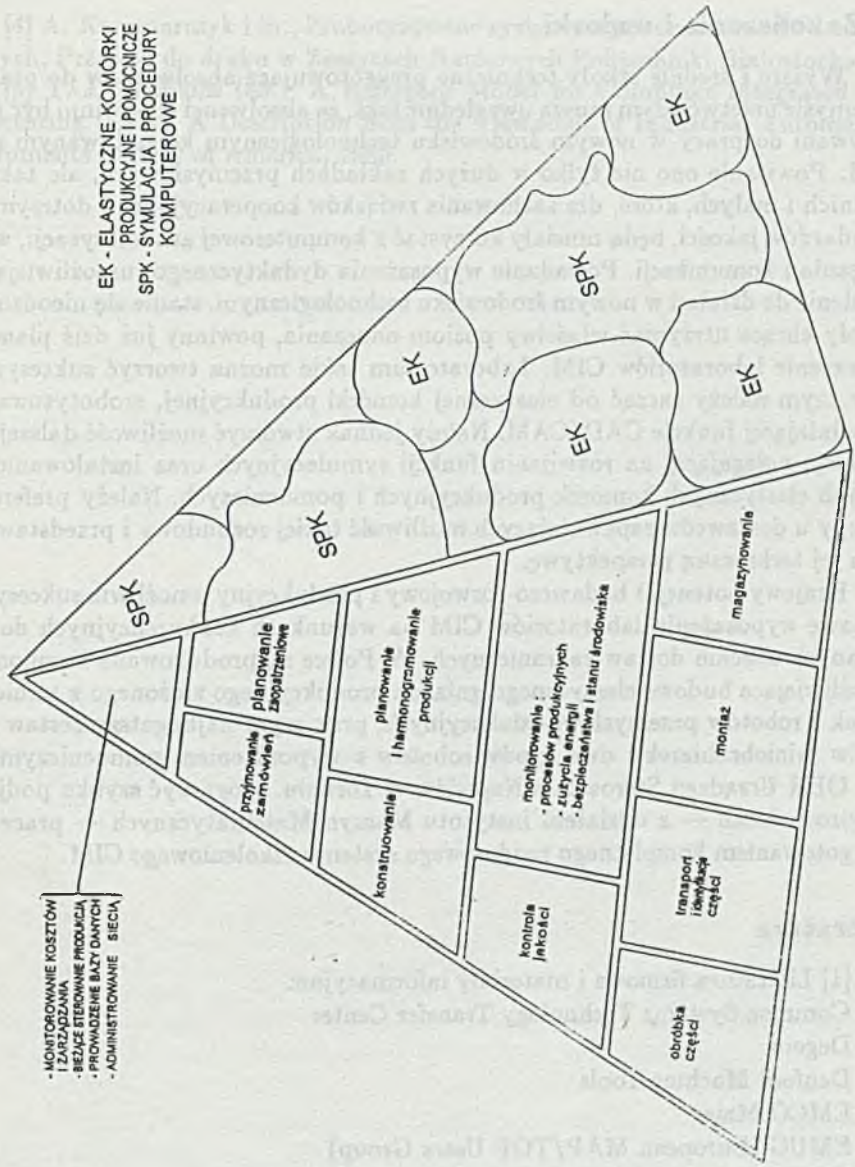
Zasada modułowej budowy stanowiska pozostanie niezmieniona. Moduły będą miały charakter elastycznych komórek wytwórczych i pomocniczych, dających się łączyć, w różnych konfiguracjach, w szkoleniowy system wytwórczy. Powinna istnieć możliwość zastępowania fragmentów systemu komputerową symulacją ich funkcji. Taką koncepcję stanowiska szkoleniowego ilustruje Rys. 6.

Poszczególne moduły, jak i w obecnych rozwiązaniach, będą mogły też być używane jako samodzielne stanowiska dydaktyczne. Sterownikami modułów będą przede wszystkim komputery osobiste, połączone standardową siecią lokalną; używane będą również sterowniki programowo-logiczne.

Stanowiska szkoleniowe budowane z różnych modułów, o różnym wyposażeniu technologicznym i programowym, będą mogły służyć do produkcji różnorodnych wyrobów, zależnie od potrzeb i inwencji użytkownika. Jednak dostawca systemu/stanowiska powinien zapewnić możliwość wytwarzania pewnej wzorcowej grupy wyrobów (i wypełniania dla tej produkcji głównych funkcji CIM) przy określonej, zaproponowanej przez siebie, konfiguracji stanowiska i wyposażeniu modułów.

Dobrym przykładem produkcji wzorcowej jest wytwarzanie, przez zintegrowany system służący do demonstracji możliwości współpracy komputerowych środ-

* w zakresie MRP — Material/Manufacturing Resource Planning



Rys. 6. Koncepcja funkcji i struktury stanowiska szkoleniowego CIM

ków automatyzacji pochodzących od różnych wytwórców, plakietek z nazwiskami osób zwiedzających ekspozycję w *The Common Systems Technology Transfer Center*, Warren, Michigan, USA. Właśnie plakietki i znaczki z napisami i prostą grafiką, dostosowane do nich podstawki, ramki itp. oraz wyroby złożone z takich elementów, mogłyby stanowić jedno z wcieleń owej grupy wzorcowej produktów szkoleniowego systemu CIM.

4. Zakończenie i wnioski

Wyższe i średnie szkoły techniczne przygotowujące absolwentów do pracy w przemyśle przetwórczym muszą uwzględnić fakt, że absolwenci ci powinni być przygotowani do pracy w nowym środowisku technologicznym kształtowanym przez CIM. Powstanie ono nie tylko w dużych zakładach przemysłowych, ale także w średnich i małych, które, dla zachowania związków kooperacyjnych i dotrzymania standardów jakości, będą musiały korzystać z komputerowej automatyzacji, wspomagania i komunikacji. Posiadanie wyposażenia dydaktycznego, umożliwiającego szkolenie do działań w nowym środowisku technologicznym, stanie się nieodzowne. Szkoły chcące utrzymać właściwy poziom nauczania, powinny już dziś planować utworzenie laboratoriów CIM. Laboratorium takie można tworzyć sukcesywnie, przy czym należy zacząć od elastycznej komórki produkcyjnej, zrobotyzowanej i wypełniającej funkcje CAD/CAM. Należy jednak stworzyć możliwość dalszej rozbudowy, polegającej na rozwijaniu funkcji symulacyjnych oraz instalowaniu kolejnych elastycznych komórek produkcyjnych i pomocniczych. Należy preferować zakupy u dostawców zapewniających możliwość takiej rozbudowy i przedstawiających jej techniczną perspektywę.

Krajowy potencjał badawczo-rozwojowy i produkcyjny umożliwia sukcesywną dostawę wyposażenia laboratoriów CIM na warunkach konkurencyjnych do oferowanych obecnie dostaw zagranicznych. W Polsce są produkowane komponenty umożliwiające budowę elastycznego gniazda produkcyjnego złożonego z miniobrabiarek i robotów przemysłowo-edukacyjnych, przy czym najbogatszy zestaw dwu typów miniobrabiarek i dwu typów robotów z wyposażeniem pomocniczym oferuje OBR Urządzeń Sterowania Napędów w Toruniu. Mogą być szybko podjęte i przeprowadzone — z udziałem Instytutu Maszyn Matematycznych — prace nad przygotowaniem kompletnego modułowego systemu szkoleniowego CIM.

Literatura

[1] Literatura firmowa i materiały informacyjne:
Common Systems Technology Transfer Center
Degem
Denford Machine Tools
EMCO Maier
EMUG (European MAP/TOP Users Group)
IEEE (Institute of Electrical and Electronics Engineers)
IFR (International Federation of Robotics)
ISO (International Standards Organization)
Lucas Nülle
OBR Urządzeń Sterowania Napędów
TQ International
Universität Erlangen-Nürnberg

[2] L. Cronjäger (ed.), Bausteine für die Fabrik der Zukunft. Eine Einführung in die rechenintegrierte Produktion (CIM), Springer V., 1990.

[3] A. Kaczmarczyk, M. Kacprzak, Sieci komunikacyjne MAP i TOP, Zeszyty Naukowe Politechniki Białostockiej, 1992. Nauki Techniczne Nr 84. Elektryka Z. 12.

[4] A. Kaczmarczyk i in., Zrobotyzowane systemy wytwórcze do celów edukacyjnych, Przyjęte do druku w Zeszytach Naukowych Politechniki Białostockiej.

[5] T. J. Williams (ed.), A Reference Model for Computer Integrated Manufacturing (CIM). A Description from the Viewpoint of Industrial Automation, Instruments Society of America, 1989.

HANNA KUŹNICKA
INSTYTUT MASZYN MATEMATYCZNYCH WARSZAWA
JAN RYŻKO
INSTYTUT MASZYN MATEMATYCZNYCH WARSZAWA

Projekt nowych standardów dla zestawu znaków w przetwarzaniu i przesyłaniu informacji

The Draft of New Standards for Set of Characters in Information Processing and Interchanging

Streszczenie

W artykule przedstawiono problemy kodowania znaków dla przetwarzania i przesyłania informacji. W szczególności omówiono projekt ISO/IEC 10646, a dokładniej jego pierwszą dotychczas opracowaną część o nazwie „Architektura i podstawowa płaszczyzna wielojęzykowa". Zwrócono również uwagę na polskie znaki narodowe w tym projekcie. W końcowej części artykułu omówiono pierwsze numery pisma „Universe of Character" poświęconego tym zagadnieniom.

Abstract

In the paper some problems of characters coding for information processing and interchanging are shown. Especially the draft of The International Standard numbered 10646 is described which the first part called "Architecture and Basic Multilingual Plane" has been already designed. Attention is paid to Polish national characters in that draft. In the last part of the paper the first few numbers of new journal called "Universe of Character" which is commemoriated to these problems are discussed.

Do roku 1987 normy z zakresu przetwarzania danych opracowywane były w ramach ISO (the International Organization for Standardization — Międzynarodowa Organizacja Normalizacyjna), ściślej komitetu technicznego TC 97 tej organizacji. W roku 1987 do działalności w tej dziedzinie włączyła się IEC (The International Electrotechnical Commission — Międzynarodowa Komisja Elektrotechniczna) i powołany został połączony komitet ISO i IEC pod nazwą ISO/IEC Joint Technical Committee o symbolu JTC1 o zakresie tematycznym Information Technology (Technika informatyczna). W ramach tego komitetu działa kilkanaście podkomitetów, wśród których podkomitet SC2 o nazwie Character Sets & Information Coding (Zestawy znaków i kodowanie informacji) zajmuje się interesującymi nas zagadnieniami. W podkomitecie SC2 aktualnie działają dwie grupy robocze; pierwsza z nich o symbolu WG2 o nazwie Universal Multiple-Octed Coded Character Set (uniwersalny wielooktetowy zestaw kodowanych znaków) zwany krótko USC

(Universal Character Set), natomiast druga, oznaczona jako WG3 nazywa się 7-Bit and 8-Bit Codes (kody 7-mio i 8-mio bitowe) i zajmuje się dotychczas stosowanymi kodami.

Pierwszą normą znaków komputerowych był kod znany powszechnie jako ASCII (American Standard Code for Information Interchange) przyjęty jako norma ISO 646. Był to kod 7-bitowy określający 128 znaków, z których 32 to znaki funkcyjne, a 94 — znaki graficzne (alfanumeryczne, interpunkcji, nawiasy itp.). W kodzie 8-bitowym, pierwsze 128 znaków odpowiada zestawowi kodu 7-bitowego, a pozostałe 128 zajmują litery narodowe oraz graficzne znaki specjalne.

Duże firmy komputerowe wprowadzają własne zestawy znaków zwane „code pages" (strony znaków). Najbardziej znanymi takimi stronami są IBM Latin-1 oznaczona numerem 850 i Latin-2 (z literami alfabetów Europy centralno-wschodniej) oznaczona jako 852, a także Microsoft Windows 1250, z których każda zapewnia kilkadziesiąt użytecznych znaków. Jednakże jest to wciąż niewystarczające dla kodów „uniwersalnych", w których przewiduje się następujące grupy znaków:

- zestaw około 220 symboli optymalizowany dla tekstów z dziedziny handlu, organizacji przedsiębiorstw i techniki,
- dodatkowe symbole stosowane w specjalnych aplikacjach wydawniczych,
- bardzo szeroki zestaw do specjalizowanych zastosowań matematycznych,
- inne znaki specyficzne dla poszczególnych alfabetów.

Poniżej przedstawione zostaną założenia projektu opracowanego przez grupę roboczą WG2 noszącego taką samą nazwę jak przytoczona wcześniej nazwa tej grupy, któremu przyporządkowano numer 10646. Korzystano tu z dokumentu o symbolu ISO/IEC DIS 10646-1.2 z 26 grudnia 1991 roku ([1]).

Dotychczas opracowana część pierwsza tego projektu zatytułowana jest „Architektura i podstawowa płaszczyzna wielojęzykowa". Zawarte są w niej określenia podstawowych pojęć wykorzystywanych w tej normie, określenie ogólnej struktury zestawu kodowanych znaków i sprecyzowanie wspomnianej w tytule podstawowej płaszczyzny wielojęzykowej (Basic Multilingual Plane — BMP). Ponadto podane jest tu określenie znaków graficznych używanych w alfabetach światowych, wyszczególnienie reprezentacji kodowych i nazw znaków graficznych BMP, sprecyzowanie 4-oktetowej formy kanonicznej UCS oznaczonej UCS-4 i dwuoktetowej formy (UCS-2), a także określenie reprezentacji kodowych dla funkcji sterowania i zarządzania przyszłymi uzupełnieniami do tego zestawu.

Wymagania zgodności, jakie tu są stawiane, odnoszą się do wymiany informacji oraz urządzeń, jakie tu mogą wystąpić. Pierwszy rodzaj wymagań jest spełniony, gdy elementy danych w postaci znaków kodowanych spełniają określenia definicji znaku i jego granic, funkcji sterowania i położenia w tablicy kodowej. Natomiast drugi rodzaj wymaga opisu urządzenia, które służy do wprowadzania, przesyłania lub odbierania znaków.

Odnośnymi dokumentami związanymi z tym projektem są: wspomniana norma ISO 646, ISO 2022, mówiąca o 7 i 8-bitowych zestawach znaków kodowanych i technikach rozszerzania kodu, ISO 6937 poświęcona zestawowi kodowanych znaków graficznych do transmisji tekstu. ISO 8859 o zestawie znaków graficznych kodowanych jednym oktetem oraz ISO/IEC 6429, mówiąca o funkcjach sterujących zestawów znaków kodowanych.

Prace nad przygotowaniem projektu ISO/IEC 10646 rozpoczęły się jesienią 1984 roku początkowo z udziałem siedmiu krajów (W. Brytania, Francja, Japonia, Niemcy, Szwecja, USA i były ZSRR). Potem brało w nich udział 15 krajów i szereg zainteresowanych organizacji takich jak ECMA (European Computer Manufacturers Association), SC 18/WG 8, TC 46 i CCITT (po reorganizacji — UIT-TS), łącznie około 30 uczestników. Pierwszy projekt normy międzynarodowej (DIS) opracowany został w grudniu 1990 roku. Wówczas dopuszczono szereg form reprezentacji znaków mających cztery lub więcej oktetów. Nie można było kodować tylko znaków graficznych w miejscach zwyczajowo przyjętych dla znaków poleceń. Poza tym, przewidywano oddzielne ćwierćpłaszczyzny dla trzech narodowych norm Chin, Japonii i Korei. Później przyjęto zunifikowany repertuar zawierający te trzy normy łącznie ze znakami chińskimi z Tajwanu i Hong-Kongu.

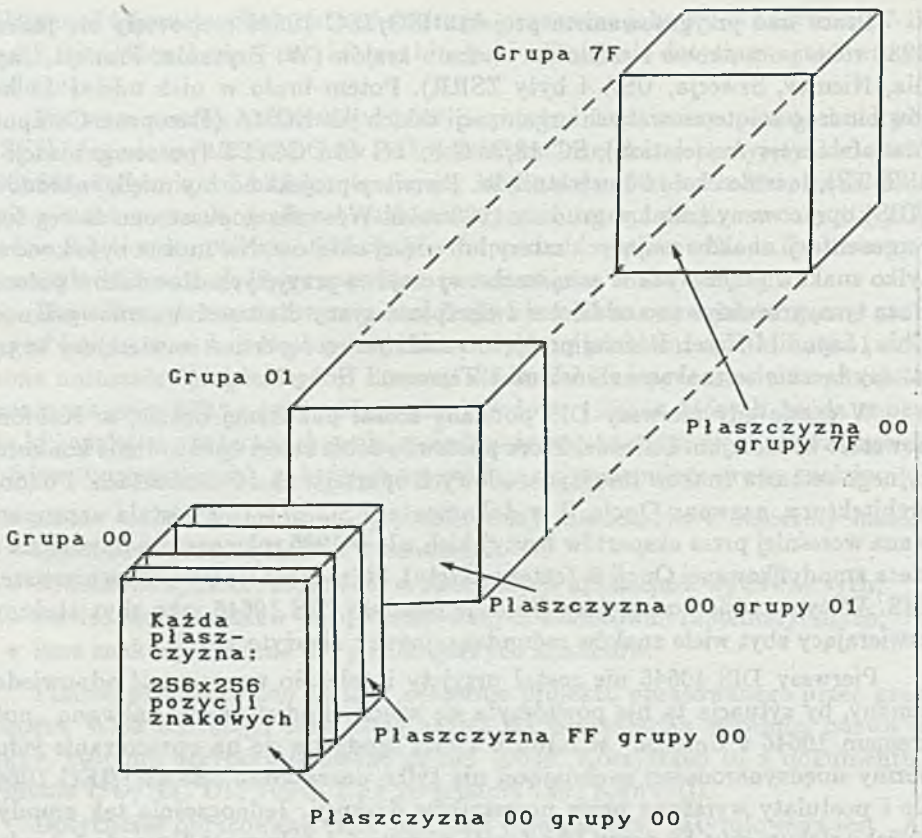
W czasie gdy pierwszy DIS poddany został publicznej ocenie, w Kalifornii powstało konsorcjum Unicode, które postawiło sobie za cel opracowanie konkurencyjnego zestawu znaków międzynarodowych opartego na 16 elementach. Podobna architektura, nazwana Opcją B w dokumentach SC 2/WG 2 została zaproponowana wcześniej przez ekspertów brytyjskich, ale w 1986 roku została odrzucona na rzecz zmodyfikowanej Opcji A (cztery oktety), która stała się podstawą pierwszego DIS. W tym czasie konsorcjum Unicode oceniało DIS 10646 jako zbyt złożony i zawierający zbyt wiele znaków redundancyjnych i nieużytecznych.

Pierwszy DIS 10646 nie został przyjęty i należało wprowadzić odpowiednie zmiany, by sytuacja ta nie powtórzyła się więcej. Podjęto akcję nazwaną „połączeniem 10646 z Unicode" w ramach której zgodzono się na opracowanie jednej normy międzynarodowej spełniającej nie tylko początkowe cele ISO/IEC 10646, ale i postulaty wyrażane przez uczestników dyskusji. Jednocześnie tak zmodyfikowano dokumenty, by nowy DIS (część pierwsza) i dokument Unicode 1.1 miały ten sam zestaw znaków. W grudniu 1991 roku przedstawiono drugi DIS, który spotkał się z dużym uznaniem organizacji narodowych. Na specjalnym spotkaniu komitetu redakcyjnego JTC 1/SC 2/WG 2 w czerwcu 1992 roku w Seulu wprowadzono tylko drobne poprawki. Opracowanie projektu było niezwykle pracochłonne i czasochłonne. Uwidoczniła się tu znacząca rola środków telekomunikacji w takich przedsięwzięciach.

Ogólna struktura omawianego zestawu kodowanych znaków przedstawiona jest na Rys. 1 i 2. Wartość każdego oktetu wyrażona jest w zapisie szesnastkowym (heksadecymalnym) od 00 do FF. Forma kanoniczna tego zestawu kodowanych znaków wykorzystuje czterowymiarową przestrzeń kodowania, rozpatrywaną jako jedna całość, składającą się z 128 grup trójwymiarowych. W ten sposób najbardziej znaczący (ósmy) bit bajtu w formie kanonicznej kodowanego znaku może być wykorzystany do celów wewnętrznego przetwarzania w urządzeniu, dopóki jest on ustawiony na zero w ramach odpowiedniego elementu danych kodowanego znaku.

Każda grupa składa się z 256 płaszczyzn 2-wymiarowych, każda płaszczyzna składa się z 256 jednowymiarowych rzędów (wierszy), a każdy rząd zawiera 256 komórek. Znak jest umieszczony i zakodowany w komórce w ramach tej przestrzeni kodującej lub komórka jest opisana jako niewykorzystana.

W formie kanonicznej do reprezentowania każdego znaku wykorzystane są cztery oktety i one określają odpowiednio: grupę, płaszczyznę, rząd i komórkę.



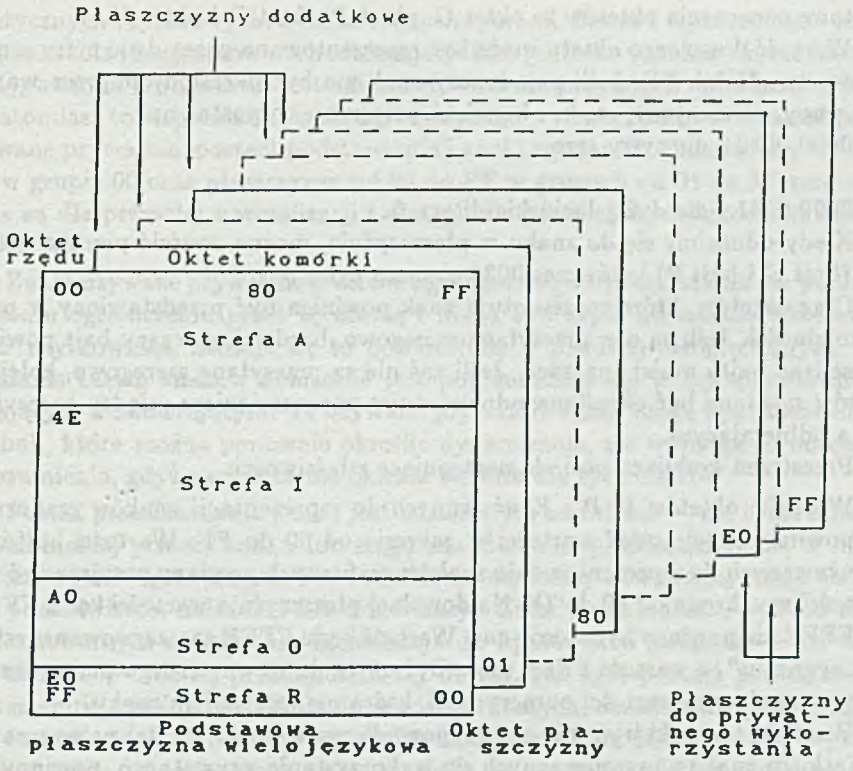
Rys. 1. Cała przestrzeń kodowania uniwersalnego wielooktetowego zestawu kodowanych znaków

Dzieje się tak dlatego, że dwa oktety nie wystarczałyby do objęcia wszystkich alfabetów światowych, a ponadto 32-bitowa reprezentacja odpowiada architekturze współczesnych procesorów.

Pierwsza płaszczyzna (płaszczyzna 00 grupy 00) nazwana jest podstawową płaszczyzną wielojęzykową (BMP). Zawiera ona znaki ogólnie używane w pismach (ang. *scripts*) alfabetycznych, sylabowych i ideograficznych łącznie z różnymi symbolami i cyframi. BMP posiada strefę ograniczonego wykorzystania, w której znaki mają specjalne właściwości. Kolejne płaszczyzny traktowane są jako dodatkowe i będą one zawierać dodatkowe znaki graficzne.

32 płaszczyzny o wartościach oktetu płaszczyzny od E0 do FF grupy 00 przeznaczone są do prywatnego wykorzystania. 32 grupy, o wartościach oktetu grupy 60 do 7F tego zestawu kodowanych znaków, są również przeznaczone do prywatnego wykorzystania. Zawartości komórek w strefach prywatnego wykorzystania nie są wyszczególniane w tym projekcie. Każdy znak określony jest w zestawie kodowanych znaków poprzez oktet grupy, oktet płaszczyzny, oktet rzędu i oktet komórki.

Oprócz postaci kanonicznej określona jest również postać dwuoktetowa. W ten sposób podstawowa płaszczyzna wielojęzykowa może być traktowana jako dwuok-



Rys. 2. Grupa 00 uniwersalnego wielooktetowego zestawu kodowanych znaków

tetowy zestaw kodowanych znaków określany jako USC-2. Podzestawy przestrzeni kodowania mogą być wykorzystane do określenia podzestawów znaków graficznych.

Jak już wspomiano, uniwersalny wielooktetowy zestaw znaków kodowanych traktowany jest jako jedna całość zawierająca 128 grup, z których każda złożona jest z 256 rzędów znaków, a każdy rząd zawiera 256 komórek.

Na tablicy kodowej przedstawiającej zawartość płaszczyzny (Rys. 2) oś pozioma reprezentuje najmniej znaczący oktet o mniejszych wartościach z lewej strony, a oś pionowa — bardziej znaczący oktet o mniejszych wartościach od góry.

Każda oś tej przestrzeni kodowana będzie za pomocą jednego oktetu. W ramach każdego oktetu najbardziej znaczącym będzie bit ósmy, a najmniej znaczącym — pierwszy.

Kodowanie znaków odbywa się w ten sposób, że każdy znak w kanonicznej formie zestawu kodowanych znaków reprezentowany jest przez ciąg czterech oktetów. Najbardziej znaczący oktet to oktet grupy, a najmniej znaczący to oktet komórki. Ciąg ten może więc być przedstawiony jako:

nb. z. nm. z.

Oktet grupy	Oktet płaszczyzny	Oktet rzędu	Oktet komórki

gdzie nb. z. znaczy oktet najbardziej znaczący, a nm. z. — najmniej znaczący.

Skrótowe oznaczenia oktetów to oktet G, oktet P, oktet R i oktet K.

Wartość dowolnego oktetu może być reprezentowana przez dwie cyfry szesnastkowe, np. 31 lub FE. Jeśli pojedynczy znak ma być określony poprzez wartość jego grupy, płaszczyzny, rzędu i komórki, przyjmuje to postać np.

0000 0030 dla cyfry zero,

lub

0000 0041 dla dużej łacińskiej litery A.

Kiedy odnosimy się do znaku w płaszczyźnie, można opuścić pierwsze cztery zera (bajt G i bajt P) i wówczas 0030 oznacza cyfrę zero.

Ciąg oktetów, które reprezentują znak powinien być przedstawiony w podanej kolejności. Jeśli są one przesyłane szeregowo, bardziej znaczący bajt powinien poprzedzać bajty mniej znaczące. Jeśli zaś nie są przesyłane szeregowo, kolejność oktetów powinna być określona odpowiednim porozumieniem między przesyłającym a odbierającym.

Przestrzeń kodująca posiada następujące właściwości:

1. Wartości oktetów P, R i K używanych do reprezentacji znaków graficznych powinny przyjmować wartości w zakresie od 00 do FF. Wartości bajtów G używanych do reprezentowania znaków graficznych powinny przyjmować wartości w zakresie od 00 do 7F. Na dowolnej płaszczyźnie wartości kodu FFFE i FFFF nie powinny być stosowane. Wartość kodu FFFE zarezerwowana jest dla „sygnatur", a wartość FFFF może być użyta do wewnętrznego przetwarzania wymagającego wartości numerycznej, która nie jest kodem znaku.
2. Wartości kodu, którym nie są przyporządkowane znaki wg. tej normy, za wyjątkiem znaków przeznaczonych do wykorzystania prywatnego, powinny być zarezerwowane do normalizacji w przyszłości i nie powinny być wykorzystywane do żadnych innych celów. Pozycje kodu dla znaków do wykorzystania prywatnego nie mogą być przyporządkowane znakom graficznym w przyszłych wydaniach tej normy.
3. Ten sam znak graficzny nie może być przyporządkowany więcej niż jednej pozycji kodu. Istnieją w zestawie kodowanych znaków znaki graficzne o podobnych kształtach; stosowane są one do różnych celów i mają różne nazwy.

Jednakże może się zdarzyć, że znak końcowy jest kombinacją znaku podstawowego i np. określonego akcentu; wówczas ma inną reprezentację kodową niż znak podstawowy.

Podstawowa płaszczyzna wielojęzykowa (Rys. 2) podzielona jest na cztery strefy:

strefa A pozycje kodowe od 0000 do 4DFF (19903 pozycje z czego 11892 przyporządkowane, a 8011 rezerwowych),

strefa I pozycje kodowe od 4E00 do 9FFF (20992 pozycje z czego 20902 przyporządkowane, a 90 rezerwowanych dla przyszłych zastosowań),

strefa O pozycje kodowe od A000 do DFFF (16384 pozycje),

strefa R pozycje kodowe od E000 do FFFD (8190 pozycji z caego 6400 do zastosowań specjalnych, 1374 przyporządkowane, a 416 do przyszłych zastosowań).

Pozycje kodowe od 0000 do 001F i od 007F do 009F w BMP zarezerwowane są dla znaków sterowania. Pozostałe pozycje strefy A wykorzystane są do pism al-

fabetycznych i sylabowych, a także różnych symboli. Strefa I wykorzystana jest do zunifikowania ideogramów wschodnioazjatyckich (chińsko-japońsko-koreańskich — CJK), a strefa O (otwarta) jest zarezerwowana dla przyszłych standardów. Strefa R natomiast to wspomiana strefa ograniczonego wykorzystania zawierająca znaki używane prywatnie, postaci podstawiania i znaki zgodności. Płaszczyzny od 01 do DF w grupie 00 oraz płaszczyzny od 00 do FF w grupach od 01 do 5F zarezerwowane są dla przyszłej normalizacji i dlatego nie powinny być wykorzystywane do innych celów.

Znaki używane prywatnie w strefie ograniczonego wykorzystania nie podlegają żadnemu ograniczeniu przez tę normę i mogą być użyte do znaków określonych przez użytkownika. Stosuje się to powszechnie w pismach ideograficznych. Przy wymianie takich znaków konieczne jest porozumienie się w tej sprawie między nadającym a odbierającym. Te używane prywatnie znaki mogą być stosowane do symboli, które można ponownie określić dynamicznie, ale wymaga to odrębnego porozumienia, gdyż norma ta nie określa technik dla tych znaków.

Postać przedstawienia znaku jest odmiennym kształtem — nie tylko odmianą — nominalnej postaci znaku lub ciągu znaków jakie przedstawione są w innych strefach znaków graficznych. Przekształcenie od formy nominalnej może obejmować podstawienie, nałożenie lub kombinację. Zasady nakładania, wyboru różnie ukształtowamych znaków lub kombinacja do ligatur albo połączeń — co często jest złożone — nie są przedmiotem tej normy. Na ogół postaci przedstawienia nie zastępują nominalnych postaci znaków graficznych, określonych gdzie indziej w tym zestawie znaków. Jednakże specjalne zastosowania mogą zakodować te postaci zamiast postaci nominalnych np. ze względu na zgodność z istniejącymi urządzeniami.

Znaki zgodności włączone są do tej normy głównie ze względu na to, by pozwolić na dwukierunkową konwersję kodu bez straty informacji.

Rewizji i uaktualniania tego zestawu kodowanych znaków dokonywać będzie podkomitet ISO/IEC JTC1/SC2, przy czym zakłada się, że nazwy i przyporządkowania przyjęte tutaj nie ulegną zmianie.

Norma ta przewiduje określenie podzestawów kodowanych znaków graficznych, które wykorzystywane są przy wymianie między urządzeniem inicjującym a odbierającym. Rozróżnia się podzestaw ograniczony i wybrany. Przyjęty podzestaw może być jednym z nich lub ich kombinacją. Ograniczony podzestaw składa się z listy znaków graficznych w tym określonym podzestawie. Pozwala to na stosowanie aplikacji i urządzeń wykorzystujących inne kody do współpracy z tym zestawem kodowanych znaków. Żądanie dostosowania się w przypadku podzestawu ograniczonego sprowadza się do podania nazw znaków graficznych w tym podzestawie lub ich pozycji kodowych, tak jak to jest określone w tej normie.

Wybrany podzestaw składa się z listy zbiorów znaków graficznych określonych w tej normie. Powinien on zawsze zawierać komórki od 20 do 7E rzędu 00 płaszczyzny 00 grupy 00. Żądanie dostosowania się do wybranego podzestawu jest spełnione, gdy podawane są zbiory zgodne z tą normą.

Norma ta zapewnia dwie alternatywne formy kodowanej reprezentacji znaków: dwuoktetową formę BMP i czterooktetową formę kanoniczną. Ponadto norma określa dwa poziomy implementacji. Przy pierwszym z nich element informacji w

postaci kodowanego znaku nie powinien zawierać kodowanych reprezentacji znaków kombinowanych. Natomiast na drugim poziomie implementacji taka reprezentacja znaków kombinowanych może być stosowana. Na tym poziomie ten sam znak może mieć dwie lub więcej reprezentacji kodowych.

Zestaw znaków sterujących jest taki sam jak w podanych odnośnych normach i znaki te mają być przedstawione jako ciąg jednego lub więcej oktetów. Natomiast rozszerzenia znaków sterujących wprowadzone w normie ISO-2022 nie są tu wykorzystane.

Jeśli wysyłający posługuje się tą normą, to musi ona również być znana odbierającemu. Droga, którą sposób identyfikacji jest przekazywany odbiorcy, nie jest określona w tej normie. Jednakże pewne normy dotyczące wymiany kodowanej informacji mogą pozwalać lub wymagać, by kodowa reprezentacja identyfikacji, odnosząca się do elementu informacji w postaci znaków kodowych, tworzyła część wymienianej informacji.

Rys. 3 przedstawia podstawową płaszczyznę wielojęzykową wraz z nazwami poszczególnych zestawów znaków i ich rozmieszczeniem w rzędach. Poszczególne zestawy, identyfikowane według nazw użytych na tym rysunku, są następnie przedstawiane na szczegółowych tablicach, z których trzy pierwsze są przytoczone jako Tabl. 1, 2 i 3. Pierwsza z nich to powtórzenie normy ISO-646 zawierającej nie wymienione tu znaki sterowania (w pierwszych dwóch kolumnach) oraz znaki interpunkcji, niektóre symbole matematyczne, cyfry oraz duże i małe litery łacińskie. Druga, nosząca nazwę Dodatek Łaciński-1, zawiera dwa dalsze rzędy znaków sterujących oraz dwa rzędy kolejnych symboli walut i innych znaków. Natomiast prawa połowa tablicy to różne odmiany liter alfabetu łacińskiego, w których z polskich znaków narodowych występuje tylko Ó na miejscu 211 (0D3 w kodzie szesnastkowym) i ó na miejscu 243 (0F3 w kodzie szesnastkowym). Trzecia tablica, o nazwie Rozszerzony Łaciński-A, zawiera pierwsze 128 znaków pierwszego rzędu i poza ostatnim znakiem są to rozszerzenia alfabetu łacińskiego. Znajdują się tu wszystkie pozostałe znaki polskie jak to pokazano w Tabl. 4.

Poszczególne znaki określone są w normie poprzez ich graficzny obraz, reprezentację kodową i nazwę. Symbole graficzne należy traktować jako typowe obrazowe reprezentacje graficzne tych znaków. Niektóre z nich są nieco powiększone dla poprawienia czytelności. Norma nie opisuje dokładnie kształtu każdego znaku. Na kształt wpływa projekt wykorzystywanej czcionki, co nie wchodzi w zakres tej normy. W większości przypadków znaki graficzne są jednoznacznie identyfikowane przez swe nazwy. Niektóre znaki mają taki sam kształt mimo różnych nazw, jak np. duża łacińska litera A, grecka duża litera alfa i duża litera A w cyrylicy.

Ciekawostką może być fakt, że do opisu polskich znaków narodowych, nawet w angielskim orginale normy, wykorzystuje się słowo „ogonek" przy dużych i małych literach ą i ę, natomiast dla ukośnych kresek nad ć, ń, ó, ś i ź angielskie słowo „acute", a dla kropkki nad ż „dot".

Oktet rzędu				
00	ISO-646 IRV			Dodatek do Łacińskiego 1
01	Rozszerzony Łaciński-A			Rozszerzony Łaciński-B
02	Rozszerzony Łaciński-B	Rozszerzenia IPA		Litery modyf. rozmieszcz.
03	Znaki diakrytyczne			Grecki
04	Cyrylica			
05		Armeński		Hebrajski
06	Arabski			
07–08				
09	Devanagari			Bengalski
0A	Gurmukhi			Gujarati
0B	Oriya			Tamil
0C	Telugu			Kannada
0D	Malajam			
0E	Tai			Lao
0F				
10	Tybetański			Gruziński
11–1D				
1E	Dodatkowy rozszerzony Łaciński			
1F	Rozszerzenia Greckiego			
20	Znaki interpunkcji	Indeksy górne/dolne	Symbole waluty	Znaki diakr. symb.
21	Symbole literopodobne	Postaci liczb		Strzałki
22	Operatory matematyczne			
23	Różne znaki techniczne			
24	Obrazki sterujące	O.C.R. (opt. rozp. znaków)		Dołączone znaki alfanum.
25	Rysowanie ramek	Elementy blokowe		Kształty geometryczne
26	Różne znaki przedmiotów użytkowych			
27	Nazwy nowych urządzeń i ich części			
28–29				
30	Symb. i znaki interp. CJK	Hiragana		Katakana
31	Bopomofo	Hangul Jamo	Różne znaki CJK	Kombinacje Hangul Jamo
32	Dołączone litery i miesiące CJK			
33	Słowa zgodności i godziny CJK		Skróty zgodności i godziny CJK	
34–3D	Hangul			
3E–45	Dodatkowy Hangul			
46–4C	Stary Hangul			
4D				
4E–9F	Zunifikowane ideogramy CJK			
A0–DF				
E0–F7	Obszar do prywatnego użytku			
F8				
F9–FA	Ideogramy zgodności CJK			
FB	Formy przedst. alfabet.			
FC–FD	Arabskie formy przedstawiania-A			
FE		Formy zgodn. CJK	Odm. małych form	Arabskie formy przedst.-B
FF	Formy o połówkowej i pełnej szerokości			Specj.

Rys. 3. Podstawowa płaszczyzna wielojęzykowa

	000	001	002	003	004	005	006	007
0	000	015	032	0 048	@ 064	P 080	` 096	p 112
1	001	017	! 033	1 049	A 065	Q 081	a 097	q 113
2	002	018	" 034	2 050	B 066	R 082	b 098	r 114
3	003	019	# 035	3 051	C 067	S 083	c 099	s 115
4	004	020	$ 036	4 052	D 068	T 084	d 100	t 116
5	005	021	% 037	5 053	E 069	U 085	e 101	u 117
6	006	022	& 038	6 054	F 070	V 086	f 102	v 118
7	007	023	' 039	7 055	G 071	W 087	g 103	w 119
8	008	024	(040	8 056	H 072	X 088	h 104	x 120
9	009	025	) 041	9 057	I 073	Y 089	i 105	y 121
A	010	026	* 042	: 058	J 074	Z 090	j 106	z 122
B	011	027	+ 043	; 059	K 075	[091	k 107	{ 123
C	012	028	, 044	< 060	L 076	\ 092	l 108	| 124
D	013	029	- 045	= 061	M 077	] 093	m 109	} 125
E	014	030	. 046	> 062	N 078	^ 094	n 110	~ 126
F	015	031	/ 047	? 063	O 079	_ 095	o 111	127

G = 00

P = 00

Tabl. 1. Rząd 00: ISO-646 IRV

	008	009	00A	00B	00C	00D	00E	00F
0	128	144	NBSP 160	° 176	À 192	Ð 208	à 224	ð 240
1	129	145	¡ 161	± 177	Á 193	Ñ 209	á 225	ñ 241
2	130	146	¢ 162	² 178	Â 194	Ò 210	â 226	ò 242
3	131	147	£ 163	³ 179	Ã 195	Ó 211	ã 227	ó 243
4	132	148	¤ 164	´ 180	Ä 196	Ô 212	ä 228	ô 244
5	133	149	¥ 165	µ 181	Å 197	Õ 213	å 229	õ 245
6	134	150	¦ 166	¶ 182	Æ 198	Ö 214	æ 230	ö 246
7	135	151	§ 167	· 183	Ç 199	× 215	ç 231	÷ 247
8	136	152	¨ 168	¸ 184	È 200	Ø 216	è 232	ø 248
9	137	153	© 169	¹ 185	É 201	Ù 217	é 233	ù 249
A	138	154	ª 170	º 186	Ê 202	Ú 218	ê 234	ú 250
B	139	155	« 171	» 187	Ë 203	Û 219	ë 235	û 251
C	140	156	¬ 172	¼ 188	Ì 204	Ü 220	ì 236	ü 252
D	141	157	SHY 173	½ 189	Í 205	Ý 221	í 237	ý 253
E	142	158	® 174	¾ 190	Î 206	Þ 222	î 238	þ 254
F	143	159	¯ 175	¿ 191	Ï 207	ß 223	ï 239	ÿ 255

G = 00
P = 00

Tabl. 2. Rząd 00: Dodatek Łaciński-1

	010	011	012	013	014	015	016	017
0	Ā 000	Đ 016	Ġ 032	İ 048	l· 064	Ō 080	Š 096	Ū 112
1	ā 001	đ 017	ġ 033	ı 049	Ł 065	ō 081	š 097	ū 113
2	Ă 002	Ē 018	Ģ 034	IJ 050	ł 066	Œ 082	Ţ 098	Ŭ 114
3	ă 003	ē 019	ġ 035	ij 051	Ń 067	œ 083	ţ 099	ŭ 115
4	Ą 004	Ĕ 020	Ĥ 036	Ĵ 052	ń 068	Ŕ 084	Ť 100	Ŵ 116
5	ą 005	ĕ 021	ĥ 037	ĵ 053	Ņ 069	ŕ 085	ť 101	ŵ 117
6	Ć 006	Ė 022	Ħ 038	Ķ 054	ņ 070	Ŗ 086	Ŧ 102	Ŷ 118
7	ć 007	ė 023	ħ 039	ķ 055	Ň 071	ŗ 087	ŧ 103	ŷ 119
8	Ĉ 008	Ę 024	Ĭ 040	ĸ 056	ň 072	Ř 088	Ũ 104	Ÿ 120
9	ĉ 009	ę 025	ĭ 041	Ĺ 057	ʼn 073	ř 089	ũ 105	Ź 121
A	Ċ 010	Ě 026	Ī 042	í 058	Ŋ 074	Ś 090	Ū 106	ź 122
B	ċ 011	ě 027	ī 043	Ļ 059	ŋ 075	ś 091	ū 107	Ż 123
C	Č 012	Ĝ 028	Ĭ 044	ļ 060	Ŏ 076	Ŝ 092	Ŭ 108	ż 124
D	č 013	ĝ 029	ĭ 045	Ľ 061	ŏ 077	ŝ 093	ŭ 109	Ž 125
E	Ď 014	Ğ 030	Ī 046	I 062	Ő 078	Ş 094	Ů 110	ž 126
F	ď 015	ğ 031	ī 047	ľ 063	ő 079	ş 095	ů 111	127

G = 00
P = 00

Tabl. 3. Rząd 01: Rozszerzony Łaciński-A

Litera	Ą	ą	Ć	ć	Ę	ę	Ł	ł	Ń	ń	Ś	ś	Ź	ź	Ż	ż
Pozycja w Tabl. 3	004	005	006	007	024	025	065	066	066	067	090	091	121	122	123	124
Kod 16	104	105	106	107	118	119	141	142	143	144	15A	15B	179	17A	17B	17C

Tabl. 4. Polskie znaki narodowe w tablicy Rozszerzony Łaciński-A

Główne zasady przyjęte w DIS-2 odróżniające ten dokument od DIS-1 to:

1) przyjęcie tylko dwóch formatów (dwuoktetowego i czterooktetowego);
2) utrzymanie kodów rozkazowych normy ISO 8859;
3) usytuowanie znaków kodowych w BMP (grupa 00, płaszczyzna 00);
4) wprowadzenie trzech poziomów realizacji:
   - znaki określone w innych normach, bez znaków składanych,
   - pewne znaki składane — poziom dodany na spotkaniu w Seulu,
   - nieograniczone znaki składane — możliwość wielu reprezentacji;
5) zunifikowanie ideogramów Dalekiego Wschodu;
6) określenie nazw wszystkich znaków.

Problem uniwersalnych znaków stał się na tyle popularny, że od początku 1993 roku zaczął wychodzić w Wielkiej Brytanii dwumiesięcznik pod nazwą „Universe of Character", którego redaktorami są B. McGibbon, który przez wiele lat opracowywał oprogramowanie zwłaszcza dla systemów wydawniczych, oraz H. McG Ross, który zna alfabety wielu krajów świata. Ukazały się dotychczas (lipiec 1993) cztery pierwsze numery tego pisma. W pierwszym redaktorzy wyjaśniają sytuację w związku z istnieniem wspomianych projektów znaków komputerowych, opisują różne ich kategorie podając przykłady.

Numer ten przynosi też sprawozdanie z odbytego na początku grudnia 1992 roku w Niemczech sympozjum organizowanego przez firmę Unisys z ramienia konsorcjum Unicode. Mówiono tam o globalnym przetwarzaniu tekstów, architekturze Unicode, różnych egzotycznych alfabetach i innych interesujących uczestników tematach. M. Suignard z Microsoft Francja, opowiadał o zawiłościach systemu operacyjnego Windows NT, który będzie pierwszym wyrobem wykorzystującym Unicode jako podstawową strukturę kodowania. Pismo zawiera też stałe rubryki poświęcone nowym wyrobom (omawiana jest m.in. sprawa opóźnienia pojawienia się na rynku Windows NT) i komentarzom terminologicznym.

Drugi numer tego pisma, który ukazał się w marcu 1993 roku, przynosi nowe propozycje dołączenia nowych pism do proponowanych standardów obejmujących pismo birmańskie, wykorzystujące te same konwencje kodowania co pisma indyjskie. khmerskie, wykorzystujące również te konwencje, ale o innym rozkładzie w tablicy kodowania, oraz etiopskie, gdzie stosuje się konwencje o podziale sylab na znaki pseudospółgłosek i pseudosamogłosek, które są kodowane. Druga grupa propozycji, określonych jako wstępne, dotyczy pism Sinhala, gdzie wykorzystuje się konwencje kodowania z uprzednich propozycji brytyjskich, ale kodowanie jest inne. pism tybetańskich, znów wykorzystujące starsze propozycje brytyjskie, ale nie uwzględniające ostatnich podobnych propozycji, oraz pism mongolskich, gdzie

zastosowano nowe, eksperymentalne podejście. Ostatnia grupa propozycji dotyczy dwudziestu pięciu pism obejmujących m. in. etruski, mołdawski i staroirlandzki, gdzie występuje więcej wątpliwości niż w poprzednich grupach.

W numerze tym kontynuowane też jest przedstawianie różnych symboli jak spacje, linie poziome, znaki interpunkcji i inne, których ilość ocenia się na 210 niezbędnych w użyciu. Inny artykuł omawia rolę firmy IBM w ustalaniu standardów znaków komputerowych, m. in. jej udział w konsorcjum Unicode i w pracach nad ISO 10646.

W trzecim numerze przytaczane są doniesienia z Brukseli, gdzie dział tłumaczeń Komisji Wspólnoty Europejskiej przyjął ogólne zasady kodowania znaków komputerowych uwzględniające projekt ISO 10646. Mowa jest też o pierwszej implementacji tego projektu dokonanej w Japonii przez Masami Hasagawę z firmy DEC, stanowiącej dokument o objętości 760 stron formatu A4, w którym 78% zajmują znaki ideograficzne CKJ. Krytycznie omówiona jest wersja beta Windows NT, gdzie mimo 1690 dostępnych znaków brak jest niektórych, często używanych w Europie. Najobszerniejszy materiał numeru poświęcony jest akcentom i różnego rodzaju dodatkom stosowanym do liter łacińskich.

Ostatni (czwarty) numer pisma przynosi wiadomość o spotkaniu komitetu odpowiedzialnego za ISO 10646 w maju 1993 roku z udziałem przedstawicieli z 14 krajów, na którym omawiano sytuację w okresie publikowaia standardu i przewidywane przyszłe usprawnienia. Charakterystyczna jest duża ilość organizacji ubiegających się o status stowarzyszenia z tym komitetem. Omawiane są też w tym numerze inne publikacje na ten temat [2], jak również odnotowany jest fakt ukazania się Unicode wersji 1.0 [3] i samego standardu ISO 10646 wydanego właśnie przez British Standards Institution (754 strony z czego 70% zajmują znaki Dalekiego Wschodu).

Literatura:

[1] Information technology Universal MultipeOcted Coded Character Set (USC) — Part 1. Architecture and Basic Multilingual Pane, Working document for ISO/IEC Draft Internationa Standard 10646-1.2, December 26, 1991.

[2] M. Y. Ksar, Pour delier les languages, ISO Bulletin, Juin 1933, 2–8.

[3] The Unicode Standard — Wordwide Character Encoding, Version 1.0, v. 1. The Unicode Consortium, Addison-Wesley Publ. Comp., 1991.