АВТОМАТИЗАЦИЯ ПРИ ПРОЕКТИРАНЕТО НА СРЕДСТВА ЗА ВИЗУАЛИЗАЦИЯ В ИНФОРМАЦИОННО-УПРАВЛЯВАЩИ СИСТЕМИ

Георги Л. Георгиев
Стефан С. Стоенчев
Димитър Н. Карастоянов

ИТКР - БАН
1113 София
Г.Бончев,бл.2

В една съвременна система за управление в реално време, особено ако е изградена на основа на програмируеми контролери и мрежи от тях, визуализацията на информацията е част от тази система. Промишлените контролери са "черните кутии" на такава система. Проследяването на определен параметър или променлива при контролерите може да се наблюдава чрез развойните средства, разработени за него или чрез отделянето на определен брой слотове за реализиране на индикация. Тя бива седемсегментна ,чрез светодиоди ,мрежи от лампи, течни кристали и др.

Първият начин е средство за писане ,настройване и отлаждане на програми и в режим на непрекъсната работа той е неприложим.

Използването на втория начин изисква да се лишим от няколко слота или цял контролер в зависимост от това, какво, колко и как, искаме да наблюдаваме. Това е сравнително прост начин за визуализиране и намира приложение при по-малки системи за управление или там, където са малко на брой параметрите за визуализация. Когато се из-

ползува за визуализиране на по-голям обем и качество информация, този начин става значително по-скъп.

Най-мощното средство за визуализация на информация е видеодисплеят и не случайно компютърната графика е най-мощното средство за визуализация на мнемосхеми, технологии, процеси, графики, диаграми и изображения от най-различен характер.

Автоматизацията при проектирането на средства за визуализация на информация се осъщесвява предимно на базата на компютри IBM или съвместим с него (като Правец-16 или друг).

От двата режима на работа на видеодисплея графичният режим има безспорно предимство, тъй като дава достъп до всяка точка от екрана поотделно. При по-новите модели компютри от типа на IBM и съвместими, видеопаметта може да достигне най-много до 256K, което увеличава цветовите възможностти, а също и позволява по-голяма разрешаваща способност на графичните режими.

Визуализацията на текст в графичен режим не е нещо сложно за реализиране в сравнение с визуализирането на по-сложно графично изображение в текстов режим, тъй като текстовият режим е произлязъл от графичния. Затова и графичните системи генерират средства за визуализация обикновено в графичния режим на компютъра.

Разглежда се случаят, при който като високо ниво на една информационно-управляваща система е използуван компютър от горния тип, а като ниско ниво един или мрежа от контролери.

Автоматизацията се състои в разработването на програмна графична система за създаване и настройка на типови визуални изображения, както и проблемът за тяхното използуване в приложни информационно-управляващи системи. Графичната система използува три типа файлове:

Първият тип файлове се интерпретират от системата като статични екранни кадри, създадени или чрез програмния пакет D.Hallo или от самата система. Вторите файлове са части от екранен кадър или библиотека от екранни символи, които също са статични. Третият тип файлове са най-важните, тъй като те са ефективния продукт на графичната система. Те представляват библиотека от файлове , съдържащи данни за функционално зависими части от екрана или файлове на типови визуални изображения. Този тип изображение има динамичен характер.

Файловите функции на системата за трите типа файлове са еднакви и това улеснява програмното осигуряване. Те могат да се използуват като самостоятелни единици в някаква приложна информационно-управляваща система, като третият тип файлове винаги е зависим от първия тип.

Ще разгледаме третия тип файлове и неговото приложение в някаква приложна система, тъй като това е смисълът от използуването на графичната система.

Програмното осигуряване на една приложна информационно-управляваща система най-общо представлява една меню-организация, в която потребителят грубо казано, "се разхожда", за да наблюдава ,дистанционно да управлява или да се намесва в управляващия алгоритъм на задачата.

Някои от крайните листа на такава меню-организация съдържат информационната част на системата или екранен кадър, съществуващ като файл от първия тип. Определени части от видеодисплея са функционално зависими от състоянието на някакъв ключ или от състоянията на някакво аналогово изменение. Тези части на екрана визуално интерпретират някакъв смислов характер на тези състояния.

Целта на автоматизацията в случая е реализиране на програмен генератор за дефиниране на тези части от екрана, създаване на визуални

състояния, в които е възможно да изпадне приложната система, както и на програмен интерпретатор за тяхната настройка и използуване.

Данните, които се генерират от графичната система за една функционално зависима част на екрана, наричаме типово визуално изображени или визуална променлива. Тези данни се съдържат във файла от третия тип.

Формално форматът на една визуална променлива изглежда така:

```
-------------------------------.....-----------------
I код I дължина I P1 I P2 I P3 I .....   I PN-1 I PN I
-------------------------------.....-----------------
```

където:

код - код на операция

дължина - дължина в байтове на променливата

от P1 до PN - параметри на променливата

Другата важна част на програмната система за автоматизация е реализирането на програмен дешифратор или интерпретатор на визуална променлива. По код за определено име става стартиране на процедура, която използува параметрите на визуалната променлива за свои входни фактически параметри. Като параметри от програмния генератор, който е част от графичната система, могат да се записват онези, които се използуват и от операторите в графичен режим на езика от високо ниво, на който е написана графичната система.

Програмният интерпретатор ще стане част от приложната информационно-управляваща система и ще се стартира непрекъснато, когато приложната система се намира в екранен кадър, за който има създадени чрез графичната система файлове с визуални променливи или файлове от третия тип.

Визуалните променливи, създадени от графичната система участвуват в приложната информационно-управляваща система както всички останали променливи. Съдържанието на променливите, за които ще се интерпретира визуално-смислов характер, се приравняват към съответните визуални променливи чрез подходящ коефициент на мащабиране.

От графичната система ще може да се стартира и интерпретаторът на типови визуални изображения, като по този начин ще се симулират възможните им състояния. Целта на тази симулация е проиграване състоянията на избрана визуална променлива и евентуалната им корекция.

Мощните езици от високо ниво като Си, Паскал, Бейзик и техните "Турбо" варианти, разработени за компютрите от типа на IBM и съвместими, създадоха възможността с прости програмни средства да се визуализира информация в графичния режим на компютъра. Така например, ако се използуват двете функции за графичния режим на езика Си - "GETIMAGE" и "PUTIMAGE", може да се създадат следните варианти на визуални променливи. Формалните параметри на горните две функции, са координати на ляв горен ъгъл на екрания образ, дължина на екранния образ по X, по Y и съдържанието на образа, в случая правоъгълник или квадрат дефиниран като масив. Когато се комбинират тези параметри в смисъл, един или няколко да са функция на някакво изменение, а останалите константи, се получават следните визуални изображения.

- изменение само на съдържанието (образа):

Под съдържание се разбира цветност на програмно достъпна точка, определена като число. В този случай се променя само съдържанието на масива или са създадени чрез програмния генератор тол-

кова масива, колкото състояния би имала променливата, която искаме да визуализираме. Нейните състояния обаче ще се проявят само като изменение на цветност. Това ще бъде и визуално-смисловия характер на този тип визуално изображение. Дължината в байтове на този тип визуална променлива се изчислява по следния израз:

$$L = Kod + 1 + Xo + Yo + Xi + Yi + S * (Xi * Yi) / 4$$

където:

Kod - код на операция с дължина 1 байт;

1 - дължина на променливата - 2 байта;

Xo,Yo - координати на горния ляв ъгъл на екранния образ (по 2 байта всяка);

Xi,Yi - дължина на координатите по X и Y на образа (по два байта всяка);

S - брой състояния на образа;

Формулата е валидна за режим на видеодисплея "SCREEN 1" с 16000 байта видеопамет. При този режим в един байт се съхранява съдържанието на четири точки от екрана, съответно четири възможни цвята за всеки от тях.

- изменение само на началната координата:

Получава се движение, което може да е функция на желано изменение. Дължината в байтове е равна на:

$$L = Kod + 1 + Xi + Yi + (Xi * Yi) / 4 + S * (Xo + Yo)$$

- изменение само на дължината по X и по Y:

Получава се уголемяване или намаляване на образа. Тук неявно се променя и големината на масива (образа). Дължината в байтове е равна на:

$$L = Kod + 1 + Xo + Yo + S * (Xi + Yi + (Xi * Yi) / 4)$$

Изменя се и съдържанието, защото увеличеният или намален образ има и ново разпределение на точките.

Когато се комбинират по двойки или пък и трите параметъра да се променят като вид състояние, се получават елементи на анимация.

- изменение на съдържание и начални координати:

Дължината се пресмята по следния израз:

$$L = Kod + 1 + Xi + Yi + S * (Xo + Yo + (Xi * Yi) / 4)$$

Визуалният ефект на този тип визуална променлива е движение на някаква част от дисплея с дължини Xi , Yi и промяна на цветността му като състояние. Броят на движенията е равен на броя на промените на цветността(съдържанието).

$$S = S1 = S2$$

където,

S1 - брой движения

S2 - брой на промените на цветността

При така написания израз за дължината в байтове, визуалното изображение(променлива) е функция на един параметър.

$$V = F(S)$$

В общия случай дължината на този тип визуална променлива се пресмята по следния начин.

$$L = Kod + 1 + Xi + Yi + S1 * (Xo + Yo) + S2 * (Xi * Yi) / 4$$

От израза се забелязва, че визуалната променлива е функция на двата параметъра S1 и S2.

$$V = F(S1,S2)$$

където:

$$S1 <> S2$$

– изменение на съдържание, начални координати и големина:

Дължината е равна на:

$$L = Kod + 1 + S * (Xo + Yo + Xi + Yi + (Xi * Yi) / 4)$$

$$S = S1 = S2 = S3$$

където:

S3 – брой уголемявания/намалявания

Броят на движенията е равен на броя на промените на цветността и броя на уголемяванията/намаленията. Ето защо и тук визуалното изображение е функция на един параметър.

$$V = F(S)$$

Дължината на визуалното изображение при пълна двумерна анимация се пресмята от израза:

$$L=Kod + 1 + S1 * (Xo + Yo) + S2 * (Xi + Yi + (Xi * Yi) / 4)$$

Броят на изменение на цветността на образа е равен на увеличения или намален образ, тъй като за всеки увеличен или намален образ има ново и единствено разпределение на точките (съдържанието).

Тъй като S2 = S3 , то остават само два параметъра S1 и S2.

$$V = F(S1,S2)$$

Дължината в байтове на всички описани по-горе визуални променливи се явава и дължина на входната команда за програмния интерпретатор.

За всички описани по-горе типови визуални променливи, в програмния интерпретатор ще участвува само функцията PUTIMAGE.

Използуването на масиви като параметър на състоянието е удачно за малки образи. При по-големи образи, респективно масиви и ако процесът е бърз, няма да е възможно в реално време да се местят големи блокове от данни. Един от начините да се избегне този проблем е използуването на функцията FILL (за езика Си), която оцветява до затворен контур. Като формален параметър на визуалната променлива се използува една точка, която да е в рамките на контура и номер на цвят. Този начин беше използуван и в задачата на Подстанция "Хиподрума" - кв. Иван вазов гр.София. Той е удобен за визуализация на дискретни променливи.

В най-общия случай параметрите на един видеодисплей са координати на точка и нейния цвят. Те биха могли да се задават в определена последователност съобразно визуализацията, която се изисква от някаква конкретна приложна задача. По този начин за всяка нова задача, ако има някакъв нов тип на визуализация, ще се добавя и нов тип дешифрация в програмния интерпретатор. За всички създадени типове визуални изображения е достатъчно да се генерират с редактора на графичната система фактическите им параметри. Това е и най-ценното качество на предложената система.