

(19) 대한민국특허청(KR) (12) 등록특허공보(B1)

(51) Int. Cl.⁶
G06F 3/14

(45) 공고일자 2003년07월07일

(11) 등록번호 10-0382100

(24) 등록일자 2003년04월16일

(21) 출원번호	10-1998-0048358	(65) 공개번호	특 1999-0062562
(22) 출원일자	1998년11월12일	(43) 공개일자	1999년07월26일
(30) 우선권주장	08/990,304 1997년12월15일	미국(US)	

(73) 특허권자 인터내셔널 비지네스 머신즈 코포레이션

미국 10504 뉴욕주 아몬크

(72) 발명자 데스테파노 조지 프란시스

미국 55904-6355 미네소타주 로체스터 컨트리 로드 19 에스이 7658

(74) 대리인 김성기, 송병옥, 안성탁

심사관 : 이정숙

(54) 근접포인트를 갖는 컴퓨터 디스플레이 상에서 다수의 그래픽 사용자 인터페이스 구성 요소를 조작하는 컴퓨터 시스템 및 방법

요약

본 발명에 따른 컴퓨터 시스템 및 방법은 포인터 가까이에 위치되는 근접 범위 내에 적어도 부분적으로 위치되는 윈도우를 동시에 조작하는 접근 포인터를 사용하여 다수의 윈도우 또는 유사한 그래픽 사용자 인터페이스 구성 요소를 조작한다. 윈도우는 포인터의 이동에 응답하여 동시에 이동되거나 또는 크기가 변경(resize)될 수 있다. 대안적으로, 윈도우는 포인터 가까이에 위치되는 공통 원점(common origin)으로부터 연장되는 방사상 라인을 따라 내부 및 외부로 향하는 방향 중 어느 한 방향으로 이동되거나 또는 크기가 변경될 수 있다.

대표도

도3

명세서

도면의 간단한 설명

- <1> 도 1은 본 발명에 따른 컴퓨터 시스템의 블록도.
- <2> 도 2는 도 1의 컴퓨터 시스템용 소프트웨어 환경을 예시하는 블록도.
- <3> 도 3은 본 발명에 따른 이벤트 관리자의 메인 루틴에 대한 프로그램 플로우를 예시하는 플로우차트.
- <4> 도 4는 도 3의 초기화 이동 모드 루틴의 프로그램 플로우를 예시하는 플로우차트.
- <5> 도 5는 도 3의 초기화 크기 변경 모드 루틴의 프로그램 플로우를 예시하는 플로우차트.
- <6> 도 6은 도 4, 도 5, 및 도 8에서 요구되는 실행 윈도우 경계 세그먼트 발견의 프로그램 플로우를 예시하는 플로우차트.
- <7> 도 7은 도 3의 윈도우 그립 루틴의 프로그램 플로우를 예시하는 플로우차트.
- <8> 도 8은 도 3의 포인터 이동 핸들링 루틴의 프로그램 플로우를 예시하는 플로우차트.
- <9> 도 9는 도 3의 윈도우 해제 루틴의 프로그램 플로우를 예시하는 플로우차트.
- <10> 도 10은 포인터와 복수의 윈도우가 디스플레이되는 컴퓨터 디스플레이를 예시하는 블록도.
- <11> 도 11은 포인터 모드를 이동 또는 크기 변경 포인터 모드의 선택 후 그립 범위 내에서 정의되는 윈도우의 강조를 도시하는 도 10의 컴퓨터 디스플레이의 블록도.
- <12> 도 12는 그립 범위의 동적 결정을 예시하는 도 11의 컴퓨터 디스플레이의 부분 상세도.
- <13> 도 13은 그립 모드 선택 후 그립 범위 내에서 정의되는 실행 윈도우의 강조를 도시하는 도 11의 컴퓨터 디스플레이의 블록도.
- <14> 도 14는 포인터 모드에 응답하여 실행 윈도우의 이동을 예시하되 이동 포인트 모드에서는 실행 윈도우의 원래 위치가 점선으로 도시되는 도 13의 컴퓨터 디스플레이의 블록도.
- <15> 도 15는 포인터 이동에 응답하여 실행 윈도우의 크기 변경을 예시하되 이동 크기 변경 포인터 모

드에서는 실행 윈도우의 원래 크기가 점선으로 도시되는 도 13의 컴퓨터 디스플레이의 블록도.

- <16> 도 16은 도 13에 도시되는 그림 범위에 대한 대안적인 그림 범위를 도시하는 컴퓨터 디스플레이의 블록도.
- <17> 도 17은 도 3의 메인 루틴에 의해 핸들링될 수 있는 소정의 추가적인 이벤트용 프로그램 플로우를 예시하는 플로우차트.
- <18> 도 18은 도 17의 깊이 변경 루틴의 프로그램 플로우를 예시하는 플로우차트.
- <19> 도 19는 도 17의 확장/압축 루틴의 프로그램플로우를 예시하는 플로우차트.
- <20> 도 20은 포인터 및 복수개의 윈도우가 디스플레이되는 또 다른 컴퓨터 디스플레이를 예시하는 블록도.
- <21> 도 21은 사용자 입력에 응답하여 원점으로부터 외부 방향으로 방사상으로 실행 윈도우의 이동을 예시하는 도 20의 컴퓨터 디스플레이의 블록도.
- <22> 도 22는 본 발명에 따른 추상 스택을 도시하는 컴퓨터 디스플레이의 블록도.

발명의 상세한 설명

발명의 목적

발명이 속하는 기술분야 및 그 분야의 종래기술

- <23> 본 발명은 일반적으로 컴퓨터 시스템과 이 컴퓨터 시스템용 그래픽 사용자 인터페이스 환경에 관한 것이다. 좀 더 구체적으로, 본 발명은 윈도우의 사용자 조작과 그래픽 사용자 인터페이스 환경의 다른 사용자 인터페이스 제어에 관한 것이다.
- <24> 컴퓨터가 더욱 복잡해지고 성능이 우수해짐에 따라, 최종 사용자 또는 오퍼레이터가 컴퓨터와 인터페이스하는 방법이 매우 중요한 문제로 대두되어 왔다. 초기 컴퓨터는 최종 사용자가 본질적으로 컴퓨터의 고유 이진 언어로 인터페이스하도록 요구했던 스위치, 광(light), 및/또는 펀치 카드에 의존했다. 결국, 영상 디스플레이는 최종 사용자가 문자형 정보와 명령(command)을 통해 컴퓨터와 인터페이스할 수 있도록 개발되었다. 이러한 발전의 또 다른 중요한 진보는 그래픽 사용자 인터페이스(graphical user interface: GUI)의 개발인데, 이 GUI는 최종 사용자가 그래픽 디스플레이 소자에서 마우스 또는 다른 사용자 인터페이스 장치에 의해 제어되는 포인터를 사용하여 "포인팅 및 클릭킹"하는 직관적인 동작을 통해 컴퓨터와 인터페이스할 수 있도록 해준다.
- <25> 정보는 흔히 윈도우로 알려져 있는 그래픽 사용자 인터페이스 구성 요소를 사용하여 GUI 환경의 사용자에게 제공된다. 간혹 윈도우는 비교적 대량의 문서 중 일부만이 임의의 주어진 시간에 보여지도록 디스플레이하는데 사용된다. 통상적으로, 사용자가 문서의 다른 부분을 볼 수 있도록 하기 위해 윈도우 상에서 문서를 검색(navigate)하는데 스크롤바 제어가 사용될 수 있다. 또한, 윈도우 상에서 다른 제어를 사용하면, 사용자는 윈도우 상에서 윈도우를 디스플레이 상의 다른 위치로 이동시키고, 경계부의 하나 이상의 세그먼트를 이동시킴으로써 윈도우의 크기를 변경시키고, 윈도우를 아이콘으로 축소하며, 윈도우를 전체 화면으로 확대하는 것 등을 포함하는 다양한 동작을 수행할 수 있다. 다이얼로그 박스 등과 같은 기타 윈도우는 상술한 제어들 중 하나 이상을 생각할 수 있다.
- <26> GUI 환경에서 실행되는 각 컴퓨터 소프트웨어 애플리케이션은 통상 하나 이상의 윈도우가 할당되어 사용자에게 정보를 제공하며/제공하거나 사용자로부터 입력을 수신한다. 또한, 다수의 컴퓨터 시스템은 멀티태스킹하는(즉, 동시에 하나 이상의 컴퓨터 소프트웨어 애플리케이션을 실행하는) 능력을 제공하여 다수의 애플리케이션으로부터 윈도우들이 동시에 컴퓨터 디스플레이에 디스플레이될 수 있다. 그 결과, 다수의 윈도우들이 임의의 주어진 시간에 컴퓨터 시스템에 디스플레이될 수 있다.
- <27> 다수의 윈도우들이 통상 겹쳐진 형태로 컴퓨터 시스템에 디스플레이되는데, 전면 윈도우는 사용자 입력 지시를 받을 컴퓨터 소프트웨어 애플리케이션과 연관되는 활성 윈도우이고, 나머지 각 윈도우는 디스플레이 상의 다른 윈도우와 비교해 얼마나 "안쪽에 있는지(deep)"를 나타내는 레벨로 표시된다. 또한, 비활성 윈도우의 임의의 겹쳐진 부분은 사용자로부터 숨겨진다. 그 다음에 비활성 윈도우가 통상적인 방법으로 활성화되고 (다른 가능한 사용자 입력이 사용될 수 있지만) 윈도우 상에서 포인팅과 클릭킹에 의해 전면으로 나오게 된다.
- <28> 사용자는 컴퓨터 태스크를 수행하는 동안 흔히 다수의 윈도우를 신속하고 또한 자주 액세스할 필요가 있다. 예를 들어, 과도한 스케줄로 인해, 사용자는 동시에 다수의 태스크에서 작업하도록 요구될 수 있는데, 예를 들어, 컴퓨터로부터 다른 태스크의 결과를 기다리는 동안 특정 태스크를 처리하는 경우를 들 수 있다. 그러나, 다수의 윈도우가 컴퓨터 디스플레이에 동시에 디스플레이되는 경우, 컴퓨터 디스플레이 상에서 사용할 수 있는 공간이 제한되어 있으므로 각 윈도우와 자주 대화하는 것이 어려울 수 있다. 윈도우는 다른 윈도우가 포인팅과 클릭킹에 의해 활성화되는 경우 다른 윈도우 배면에 완전히 숨겨지는 것이 가능하고, 그 후 다수의 사용자 대화가 없으면 사용자에게 액세스될 수 없다. 이동 및 크기 변경과 같은 동작을 수행하기 위한 제어가 윈도우 상의 특정 위치에서 정의되고 각 윈도우가 통상 개별적으로 조작되어야 하기 때문에, 상기 동작도 역시 번거롭다. 따라서, 사용자는 컴퓨터 태스크를 바로 수행하기보다는 윈도우를 조작하는데 과도한 시간을 소비할 수 있다.
- <29> 또 다른 중요한 점은 동시에 멀티 태스크를 작업하는 경우 사용자가 태스크와 윈도우 사이의 관계를 유지하는 것이 어려울 수 있다는 점이다. 윈도우와 태스크 사이의 스위칭이 혼란을 야기하여 사용자로 하여금 초점을 상실하도록 만들 수 있다.
- <30> 따라서, 그래픽 사용자 인터페이스 환경에서 윈도우 등과 같은 다중 그래픽 사용자 인터페이스

구성 요소를 조작하는 개선된 방법이 상당히 요구된다.

발명이 이루고자 하는 기술적 과제

<31> 본 발명은 다중 윈도우 또는 유사한 그래픽 사용자 인터페이스 구성 요소가 근접 포인터에 의해 조작되는 컴퓨터 시스템 및 방법을 제공함에 있어서 종래 기술과 관련된 상기 문제점과 다른 문제점을 해결하는데, 이 근접 포인터는 포인터 가까이 위치되는 근접 범위 내에 적어도 일부가 배치되는 윈도우 등을 동시에 조작한다. 본 발명의 하나의 특징에 따르면, 윈도우 등은 포인터의 이동에 응답하여 동시에 이동되거나 크기가 변할 수 있다. 본 발명의 다른 특징에 따르면, 윈도우 등은 포인터 가까이 위치되는 공통 원점(common origin)에 대해 안쪽 또는 바깥쪽 중 어느 하나로 동시에 이동되거나 크기가 변할 수 있다.

<32> 상기 어느 하나의 방법에서, 사용자는 좀 더 조직적인 방식으로 다중 구성 요소를 더욱 신속하고 용이하게 이동시키며/이동시키거나 크기가 변하도록 해준다.

<33> 본 발명의 특징을 이루는 상기 및 기타 장점과 특징들이 본 명세서에 첨부되어 본 명세서의 일부를 이루는 청구범위에 개시되어 있다. 그러나, 도면과 본 발명의 예시적인 실시예가 기술되어 있는 상세한 설명을 참조하여, 본 발명은 물론 본 발명의 사용을 통해 얻어지는 목적과 장점을 더욱 잘 이해할 수 있다.

발명의 구성 및 작용

<34> 다음에 기술되는 본 발명의 실시예에서는 사용자 조작 포인터에 대한 접근에 응답하여 윈도우 등과 같은 그래픽 사용자 인터페이스 구성 요소를 조작한다. 본 명세서에는 주로 그래픽 사용자 인터페이스 구성 요소의 조작이 디스플레이 상의 구성 요소의 계층 구조 내에서 구성 요소를 이동시키거나 크기를 변경하거나 그 상대 레벨을 변경하는 것으로 예시된다. 그러나, 그래픽 사용자 인터페이스 구성 요소의 조작이 본 발명에 따른 다른 동작을 포함할 수 있다는 것은 명백하다. 예를 들어, 적어도 윈도우 그래픽 사용자 인터페이스 구성 요소의 경우, 조작은 윈도우를 닫고, 윈도우를 최소화하거나 윈도우를 최대로 하는 것과 같은 동작을 포함할 수 있다. 또한, 조작은 접근 조작(proximity manipulation) 특성이 구현되는 특정 컴퓨터 소프트웨어 애플리케이션에 대해 특유한 다른 동작도 역시 포함할 수 있다. 또한, 그중에서도, 조작에 의해 모든 실행 구성 요소를 단일 아이콘, 메뉴 엔트리 등으로 최소화하여 최소화된 아이콘 또는 엔트리의 재저장되므로 모든 실행 구성 요소가 재저장된다. 후자의 경우는 종래 기술에서 잘 알려진 사용자 인터페이스의 한 종류인, 다중 개방 윈도우를 갖는 애플리케이션을 단일 아이콘으로 최소화하는 것과 여러 가지 면에서 유사하다.

<35> 본 명세서에 사용되는 바와 같이, 윈도우는 정보가 사용자에게 디스플레이되는 많은 그래픽 사용자 인터페이스 구성 요소 및/또는 사용자 입력이 수신되는 많은 그래픽 사용자 인터페이스 구성 요소를 포함할 수 있다. 윈도우는 하나 이상의 경계 세그먼트로 형성되는 경계를 포함한다. 또한, 하나 이상의 경계 세그먼트가 경계를 형성하는 경우, 적어도 하나의 코너가 2개의 경계 세그먼트로 이루어지는 근접 교차로 정해질 수도 있다. 통상적으로, 윈도우 경계부는 직사각형으로 배열되는 4개의 경계 세그먼트와 이들 사이에 정해지는 4개의 코너로 정의되지만, 경계 세그먼트의 기타 다른 형태(선형 또는 곡선형), 수, 및 배열이 대안적으로 사용될 수 있다. 윈도우는 스크롤 바를 포함하거나 포함하지 않을 수도 있으며, 또한 소정의 모양(예를 들어, 다이얼로그 박스와 같은) 또는 형태가 없는 것으로 정의될 수 있다. 또한, 다른 그래픽 사용자 인터페이스 구성 요소는 예를 들어, 아이콘; 텍스트 프레임, 테이블, 이미지, 아트웍(artwork) 등과 같은 데스크탑 출판 설계 요소(desktop publishing design element); 선, 박스, 원, 윤곽 등과 같은 그래픽/아트웍 요소(예를 들어, 페인팅, 도면 작업, 일러스트레이션 및 컴퓨터에 의한 설계(computer aided design: CAD 애플리케이션) 등과 같은 본 발명과 호환 가능한 동일한 방법으로 조작될 수 있다.

<36> 포인터는 화살표, 십자 표시, 또는 기타 다른 그래픽 표시를 포함하는 많은 종래의 커서(cursor) 설계로 표시될 수 있다. 또한, 특정한 포인터가 근접 조작 특성이 인에이블되는 근접 모드에 있을 때 사용될 수 있어서 모드가 인에이블되었다는 것을 사용자에게 통지된다는 것으로 이해된다. 모드의 크기 변경과 이동도 역시 개별 포인터 표시에 의해 구별될 수 있다.

<37> 본 발명의 바람직한 실시예를 기술하기 전에, 예시적인 하드웨어와 소프트웨어 환경을 간단히 설명하기로 한다.

하드웨어 환경

<38> 도면에서 동일한 도면 번호는 동일한 부분을 나타낸다. 본 발명에 따른 컴퓨터 시스템(10)이 도 1에 예시된다. 컴퓨터 시스템(10)은 네트워크(18)를 통해 서버 시스템(16)에 접속되는 하나 이상의 클라이언트 컴퓨터 시스템(12), (14) 및 (20)(예를 들어, 데스크탑 또는 개인용 컴퓨터, 워크스테이션 등)을 포함하는 네트워크 컴퓨터로 예시된다. 네트워크(18)는 실제로 네트워크 상호연결시킨 형태이며, 근거리(local-area), 광역(wide-area), 무선 및 공중 네트워크(예를 들어, 인터넷)를 포함하지만 이에 제한되는 것은 아니다. 또한, 예를 들어 다중 서버와 같은 다수의 컴퓨터와 다른 장치가 네트워크(18)를 통해 네트워크가 이루어질 수 있다. 또한, 본 발명에 따른 자립형 컴퓨터(stand-alone computer) 및 그 관련 장치에 본 발명의 원리가 사용될 수 있다는 것을 이해할 수 있다.

<39> 컴퓨터 시스템(20)은 컴퓨터 시스템(12, 14)와 유사하며, 특히 마이크로프로세서(21)과 같은 프로세서; 컴퓨터 디스플레이(22)와 같은 다수의 주변 장치; 하드 디스크, 플로피 디스크 및/또는 CD 롬 드라이브와 같은 저장 장치(23); 프린터(24); 및 다양한 입력 장치(예를 들어, 마우스(26)과 키보드(27))를 포함할 수 있다. 컴퓨터 시스템(20)은 운영 체제의 제어하에 동작하며, 다양한 컴퓨터 소프트웨어 애플리케이션, 프로그램, 객체, 모듈 등을 실행한다. 또한, 다양한 애플리케이션, 프로그램, 객체, 모듈 등을 예를 들어, 분산 컴퓨팅 환경에서 서버 또는 다른 컴퓨터 시스템(12, 14) 내의 하나 이상의 프로세서에서

실행될 수 있다.

- <41> 컴퓨터 디스플레이(22)는 사용자에게 가시적으로 정보를 제공하는 임의의 공지 방법을 포함할 수 있다. 예를 들어, 컴퓨터 디스플레이(22)는 예를 들어, 음극선관(cathode-ray tube: CRT), 액정 디스플레이(LCD), 또는 투사형(projection) 디스플레이 등과 같은 비디오 모니터일 수 있다. 또한, 홀로그래픽 탱크(holographic tank) 등과 같은 3차원 디스플레이 뿐만 아니라 3차원 디스플레이(예를 들어, 가상 현실 수신기(virtual reality headset)를 시뮬레이션하는 2차원 디스플레이를 포함하는 다른 종류의 컴퓨터 디스플레이도 사용될 수 있다.
- <42> 또한, 사용자 입력은 다른 공지 사용자 입력 장치로부터 수신될 수 있다. 예를 들어, 디스플레이 상의 포인터 제어는 트랙볼, 조이스틱, 라이트 펜, 터치 감지 패드 또는 디스플레이, 디지털화기 태블릿, 및 키보드 등에 의해 핸들링될 수 있다. 또한, 상기 다수의 장치들은 버튼, 손가락 회전(thumb wheel), 슬라이더 등과 같은 하나 이상의 사용자 제어를 포함한다. 또한, 음성 및/또는 이미지 인식 기술은 사용자에게 음성 명령어 및/또는 제스처를 제공하는데 사용되어 사용자 입력을 컴퓨터에 제공한다. 또한, 다른 사용자 인터페이스 장치가 대안적으로 사용될 수 있다.
- <43> 통상적으로, 운영 체제의 일부로서 구현되는지 또는 특정 애플리케이션, 프로그램, 객체, 명령 모듈 또는 명령 시퀀스로서 구현되는지의 여부에 상관없이 본 발명의 예시된 실시예를 구현하기 위해 실행되는 루틴은 본 명세서에서 "컴퓨터 프로그램"으로 부르기로 한다. 컴퓨터 프로그램은 네트워크된 컴퓨터 시스템(10)의 장치 또는 시스템 내에서 하나 이상의 프로세서에 의해 판독되고 실행되는 경우, 그 장치 또는 시스템으로 하여금 본 발명의 다양한 특징을 구현화하는 단계 또는 구성 요소를 실행하는데 필요한 단계를 수행하도록 한다. 또한 본 발명이 컴퓨터 시스템을 충분히 기능시키는 문맥(context)으로 설명되었으며 이하에서도 설명되지만, 본 기술 분야의 당업자는 본 발명의 다양한 실시예가 다양한 형태의 프로그램 제품으로 배포될 수 있으며, 본 발명이 배포를 실제로 수행하는데 사용되는 신호 포함 매체(signal bearing media)의 특정 형태와 관계없이 동일하게 적용됨을 명백히 알 수 있다. 신호 포함 매체의 예는 휘발성 및 비휘발성 메모리 장치, 플로피 디스크, 하드 디스크, CD롬, DVD와 같은 기록 가능한 매체와 디지털 및 아날로그 통신 링크와 같은 전송형 매체를 포함하지만, 이들에 제한되는 것은 아니다.
- <44> 본 기술 분야의 당업자는 도 1에 예시되는 양호한 환경이 본 발명에 제한되지 않는다는 것을 알 수 있다. 실제로, 본 기술 분야의 당업자는 다른 대안적인 하드웨어 환경이 본 발명의 범위를 벗어남이 없이 사용될 수 있다는 것을 알 수 있을 것이다.
- <45> 소프트웨어 환경
- <46> 도 2는 본 발명에 따른 컴퓨터 시스템(20)용 소프트웨어 환경을 예시한다. 프로세서(21)는 메모리(28)뿐만 아니라 다수의 입력 및 출력에 접속되는 것으로 예시되어 있다. 예를 들어, 사용자 입력은 마우스(26) 및 키보드(27) 등을 이용하여 프로세서(21)에 의해 수신된다. 추가적인 정보가 네트워크(18)를 통해 네트워크된 컴퓨터 시스템(10)내의 컴퓨터 시스템(20) 및 다른 컴퓨터 시스템 사이에서 전달될 수 있다. 추가적인 정보는 대량 기억 장치(23)에 저장되고/저장되거나 대량 기억 장치(23)로부터 수신될 수 있다. 프로세서(21)는 디스플레이 데이터를 디스플레이(22)로 또한 출력한다. 컴퓨터 시스템(20)은 본 기술 분야에 잘 알려져 있는 바와 같이 프로세서(21) 및 각각의 구성 요소 (18, 22, 23, 26, 27, 28) 사이에 적당한 인터페이스를 포함한다.
- <47> 컴퓨터 시스템(10)은 통상적으로 운영 체제(30)의 제어를 받으며, 그 운영 체제(30)의 구성 요소가 메모리(28)에 상주하는 이벤트 관리자(32)이다. 이벤트 관리자(32)는 통상 사용자에게 정보를 디스플레이하며 사용자로부터 입력을 수신하는 것을 핸들링하는 디스플레이 관리자(도시되지 않음)와 함께 이벤트 구동된 그래픽 사용자 인터페이스(event-driven GUI) 환경을 제공한다. 그러나, 본 발명에 따른 루틴이 이벤트 관리자가 아닌 다른 프로세스에서도 사용될 수 있다는 것을 이해할 수 있다. 예를 들어, 또한 본 발명에 따른 루틴이 운영 체제의 상위(top)에서 실행되는 다양한 컴퓨터 소프트웨어 애플리케이션에서 사용될 수 있다.
- <48> 운영 체제(30) 및 이벤트 관리자(32)가 개시에 앞서 네트워크(18) 또는 대용량 저장 장치(23)에 저장될 수 있다는 것을 이해할 수 있다. 또한 운영 체제(30) 및 이벤트 관리자(32) 각각은 예를 들어 실행 도중 상이한 시간에 임의의 메모리(28), 대용량 저장 장치(23), 네트워크(18)중 임의의 곳에 상주하거나 또는 프로세서(21)의 레지스터 및/또는 캐쉬내에 상주하는 다양한 구성 요소를 가질 수 있다.
- <49> 또한, 다른 소프트웨어 환경이 선택적으로 사용될 수 있다는 것도 이해할 수 있다.
- <50> 포인터 이동에 응답하여 윈도우 등의 근접 조작
- <51> 도 3은 도 2의 이벤트 관리자에 의해 실행되는 이벤트 관리자 루틴(30)용의 전형적인 프로그램 플로우를 예시한다. 루틴(50)은 이벤트 구동 표시에서 구현되는 것으로 도시되어 있다. 그러나, 예를 들어 절차형 또는 객체 지향형 모델과 같은 다른 프로그래밍 모델이 대안적으로 사용될 수 있다. 또한, 다음의 설명은 시스템에 사용자 입력을 제공하기 위한 마우스 및 키보드를 갖는 컴퓨터 시스템에 초점이 맞춰질 수 있다. 본 발명은 상술한 임의의 다른 사용자 인터페이스 장치에 사용될 수 있으며, 따라서 본 발명이 이러한 특정 하드웨어 구성에 제한되지 않는다는 것을 이해할 수 있다.
- <52> 루틴(50)은 (통상적으로 운영 체제(30)의 배경에서 수행되는 전용 프로세스로서) 무한 루프로 동작하는 것으로 여겨질 수 있다. 본 기술 분야에 잘 알려진 바와 같이, 루틴(50)은 블록(52)에서 이벤트를 기다리며 이벤트를 적절한 핸들링 루틴에 전달한다. 본 발명에 사용가능한 이러한 다수의 이벤트는 블록(54, 56, 58, 60, 62 및 64)에서 검출된다. 또한 컴퓨터 시스템에서 발생되지만 본 발명을 완전히 이해하는 것과는 관련이 없는 다수의 다른 형태의 이벤트를 핸들링하는 블록(66)이 도 3에 도시되어 있다.
- <53> 통상적으로, 본 명세서에 기술되는 실시예에서, 포인터는 적어도 3가지 상호 배타적인 모드중 하나에서 동작하게 된다. 첫째로, 정상 모드는 GUI 환경에서 사용자 제어형 포인터의 통상적인 동작을 표시한다. 둘째, "이동" 모드는 특정 사용자 입력이 활성화된 후 사용자에게 의해 정의되는 원점(origin point)의 그림 범위(grip span) 내의 윈도우가 포인터 또는 다른 사용자 입력의 이동에 응답하여 동시에 이동

될 수 있는 모드를 표시한다. 셋째, "크기 변경" 모드는 특정 사용자 입력의 활성화후 사용자에게 의해 정의되는 원점의 그림 범위내의 윈도우가 포인터 또는 다른사용자 입력의 이동에 응답하여 동시에 크기 변경될 수 있는 모드를 표시한다.

- <54> 3가지 포인터 모드 중 하나를 선택하는 것은 본 기술 분야에 공지된 많은 방법으로 수행될 수 있다. 예를 들어, 모드의 선택은 풀 다운 메뉴 또는 팝업 메뉴, 체크 박스 또는 무선 버튼, 툴바 버튼을 통해 수행되거나 또는 특정 키스트로크 및/또는 마우스 버튼의 조합을 사용하여 수행될 수 있다. 또한, 다른 공지된 사용자 인터페이스 제어가 이용 가능한 모드들 사이에서 토글하는데 사용될 수 있다. 또한, 선호 또는 선택 다이얼로그 박스(preference or option dialogue box)를 통해, 포인터 모드가 설정될 수 있으며/설정될 수 있거나 디폴트 포인터 모드가 정의될 수 있다. 컴퓨터 시스템 상의 특정 모드를 선택하거나 선택 해제하는 다른 방법이 대안적으로 사용될 수 있다.
- <55> 블록(54)에 도시되는 바와 같이, 이동 포인터 모드의 선택은 루틴(50)에 의해 검출될 수 있다. 상기 이벤트가 수신되는 경우, 제어는 이동 모드 루틴(100)을 초기화하기 위해 전달되어 포인터를 이동 모드로 위치시킨다. 마찬가지로, 블록(56)에 도시되어 있는 바와 같이, 루틴(50)도 역시 이벤트를 수신하여 크기 변경 모드 루틴(110)을 초기화함으로써 핸들링되는 크기 변경 포인터 모드를 선택할 수 있다. 상기 설명된 이동 및 크기 변경 포인터 모드의 선택 방법은 본 발명에 따른 상기 이벤트를 초기화할 수 있다.
- <56> 정상 포인터 모드의 선택은 블록(58)에서 검출된다. 상기 이벤트에 응답하여, 제어가 블록(68)에 전달되어 포인터 모드를 정상 모드로 설정하고, 그 다음에 블록(69)으로 전달되어 정상(비근점) 포인터를 디스플레이한다. 또한, 포인터의 디스플레이는 또한 본 기술 분야에 잘 알려져 있는 방법으로 디스플레이를 리프레시하는 것이 요구될 수 있다.
- <57> 본 명세서에 기술되는 실시예에서, 일단 포인터 모델이 이동 또는 크기 변경 포인터 모드 중 어느 한 모드로 설정되면, 근점 이동 또는 크기 변경은 통상 그림 이벤트를 초기화하며 포인터를 원하는 대로 이동시킴으로써 사용자에게 의해 수행된다. 윈도우의 실제 이동 또는 크기 변경은 포인터 이동의 검출에 응답하여 수행된다. 근점 이동 또는 크기 변경은 해제 이벤트에 의해 종료된다.
- <58> 본 명세서에 기술되는 실시예에서, 그림 및 해제 이벤트는 (협력하는 키스트로크 조합없이도) 하나 이상의 마우스 버튼을 눌러 해제시킴으로써 각각 초기화될 수 있다. 또한, 상이한 마우스 버튼 또는 동일한 마우스 버튼의 조합을 누름으로써, 근점 조작 기능의 동작이 이하에서 더욱 상세히 설명되는 몇 개의 이용 가능한 옵션간에 주문제작될 수 있다. 그러나, 본 발명의 상이한 모드 및 특징을 활성화시키는 사용자 인터페이스 메카니즘이 본 명세서에 기술되는 메카니즘과는 상이할 수 있다는 것을 이해할 수 있다. 따라서, 본 발명은 상기 기술된 메카니즘에만 제한되는 것은 아니다.
- <59> 블록(60, 62 및 64)에 도시되는 바와 같이, 그림, 포인터 이동, 및 해제 이벤트는 적절한 루틴에 의해 핸들링하기 위해 루틴(50)에 의해 수신된다. 특히, 그림 이벤트는 그림 윈도우 루틴(140)에 의해 처리되고, 포인터 이동 이벤트는 포인터 이동 핸들링 루틴(160)에 의해 핸들링되며, 해제 이벤트는 윈도우 해제 루틴(190)에 의해 핸들링된다.
- <60> 초기화 이동 모드 루틴(100)은 도 4에 더욱 상세히 예시된다. 상술한 바와 같이, 컴퓨터 시스템용 포인터는 3개의 상호 배타적인 모드 중 하나에서 사용자에게 의해 설정될 수 있다. 또한, 사용자가 포인터에 대해 현재 모드를 용이하게 식별할 수 있도록 상이한 모드의 각각에 별개의 포인터 표시를 사용하는 것이 바람직하다.
- <61> 따라서, 루틴(100)은 포인터 모드를 "이동" 모드로 설정함으로써 블록(102)에서 시작한다. 다음에, 블록(104)에서는 포인터가 이동 포인터 표시로 설정된다.
- <62> 현재 모드에 좌우되는 포인터의 표시를 구별하는데 몇 가지 상이한 방법이 사용될 수 있다. 예를 들어, 이동 포인터 표시는 이동 모드가 선택되는 경우 사용자에게 표시되는 임의의 커서 표시를 포함한다. 예를 들어, 손, 화살표, 갈고리(grappling hook), 또는 기타 다른 표시가 이동 포인터를 구별하는데 사용될 수 있다. 또한, 포인터의 현재 그림 범위는 예를 들어 접근 범위의 윤곽을 정하며/정하거나 근점 범위 내의 상이한 음영(shade) 또는 컬러를 사용하여 디스플레이의 다른 부분을 구별함으로써 포인터의 확장 근점 범위를 디스플레이하는데 사용될 수 있다. 하나의 실시예로, 근점 범위 내의 모든 화소가 약한 음영(lighter shade)으로 디스플레이됨으로써 포인터는 포인터에 대한 근점 범위를 강조하는데 스포트라이트 효과를 가질 수 있다. 상이한 포인터 표시를 구별하는데 추가적인 방법이 대안적으로 사용될 수 있다.
- <63> 일단 포인터의 표시가 설정되었으면, 실행 윈도우 경계 세그먼트 발견(find affected window boundary segment) 루틴(120)이 호출되어 어느 윈도우(구체적으로는 어느 윈도우의 경계 세그먼트)가 포인터에 대한 현재 근점 범위 내에 적어도 부분적으로 배치되는지를 결정한다. 이하에서 더욱 상세히 기술되는 바와 같이, 루틴(120)은 각각의 윈도우 및 포인터의 근점 범위 내에 속하는 윈도우의 세그먼트를 표시하는 "그립된 리스트(gripped list)"로 복귀한다. 이러한 정보는 블록(106)에서 사용되어 그림 이벤트가 사용자에게 의해 초기화되는 경우 사용자가 실행 가능한 윈도우를 식별할 수 있도록 실행 윈도우(affected window)를 강조한다. 실행 윈도우의 강조(hilight)는 통상 각 윈도우 객체 내의 플래그를 설정함으로써 수행되어 디스플레이상 상기 윈도우를 비실행 윈도우(non-affected window)와 구별하기 위한 고유 표시(unique representation)가 상기 윈도우에 대해 발생된다.
- <64> 실행 윈도우는 본 기술 분야에 공지된 많은 방법으로 강조될 수 있다. 예를 들어, 윈도우의 경계 부는 음영 또는 컬러를 통해 구별될 수 있다. 또한, 상기 윈도우의 타이틀 바도 동일한 방법으로 구별될 수 있다. 다른 실시예에서, 특정 아이콘이 실행 윈도우를 비실행 윈도우와 구별하는데 사용될 수 있다. 예를 들어, 상기 아이콘이 윈도우의 타이틀 바 내에 제공되거나 또는 별개의 아이콘이 포인터 근점 범위의 주변부(perimeter)를 갖는 윈도우의 경계 세그먼트의 각 교차부에서 제공될 수 있다. 윈도우 또는 윈도우의 세그먼트를 구별하는 방법이 본 발명에 따라 사용될 수 있다.
- <65> 실행 윈도우가 강조된 후, 제어는 블록(108)으로 전달되어 디스플레이를 갱신한다. 그 결과, 포

인터를 이동 포인터 표시로 리프레시하고 각 실행 윈도우를 리프레시하여 그 실행 윈도우의 실행 상태를 디스플레이하게 된다. 통상, 디스플레이의 갱신 또는 디스플레이 상에 표시되는 구성 요소를 리프레시하는 것은 본 기술 분야에 잘 알려져 있다는 것을 이해할 수 있다.

<66> 도 5는 초기화 크기 변경 모드 루틴(110)을 좀 더 상세히 예시한다. 루틴(110)은 블록(112)가 포인터 모드를 "크기 변경" 모드로 설정하며 블록(114)이 포인터를 "크기 변경" 포인터 표시로 설정하는 것을 제외하고는 도 4의 루틴(100)과 실질적으로 동일한 방법으로 동작한다. 루틴(120)도 마찬가지로 포인터의 현재 근접 범위에 대한 실행 윈도우 경계 세그먼트를 결정하도록 호출되며, 블록(116) 및 (118)은 블록(106 및 108)과 동일한 방법으로 동작하여 실행 윈도우를 강조하고 디스플레이를 갱신한다.

<67> 크기 변경 포인터 표시가 이동 포인터 표시의 크기 변경과 동일한 방법일 수 있다는 것을 이해할 수 있다. 대안적으로, 사용자가 2가지 모드를 용이하게 구별할 수 있도록 다른 포인터 표시가 이동 및 크기 변경 포인터에 대해 사용될 수 있다.

<68> 도 6은 루틴(120)을 좀 더 상세히 예시한다. 상술한 바와 같이, 루틴(120)은 현재 근접 범위 내에 통상 적어도 부분적으로 배치되는 윈도우의 경계 세그먼트의 위치를 정한다.

<69> "근접 범위"는 포인터에 대한 근접 범위의 특성을 변경하기 위해 통상 사용자에게 의해 주문제작될 수 있는 "그립 범위(grip span)"에 좌우된다. 가장 기본적인 구성에서, 그립 범위 및 그에 따른 포인터 둘레의 근접 범위는 포인터에 의해 표시되는 중심이 현재 위치에 존재하는 미리 정해진 반경을 갖는 원으로 표시된다. 이러한 구성에서, 적어도 부분적으로 원형 영역내에 배치되는 윈도우의 임의의 경계 세그먼트는 포인터에 의해 실행되는 것으로 간주된다.

<70> 다수의 추가적인 특징이 본 발명에 따른 근접 범위로 구현될 수 있다. 예를 들어, 근접 범위의 원점은 포인터에 가까운 다른 위치에서 정의될 수 있다. 예를 들어, 근접 범위의 원점은 가장 가까운 윈도우 상의 위치(예를 들어, 가장 가까운 경계 세그먼트 상의 위치 또는 2개의 경계 세그먼트)에 의해 정의되는 코너 내의 위치로 설정될 수 있다.

<71> 또한, 근접 범위의 반경은 포인터의 현재 위치에 대해 가장 가까운 윈도우에 기초하여 동적으로 생성될 수 있다. 예를 들어, 일실시예에서, 접근 범위의 반경은 가장 가까운 윈도우 구성 요소에 스케일링 팩터(scaling factor)를 곱한 값에 대한 거리로 설정된다. 통상, 스케일링 팩터는 1보다 커서 근접 범위는 본질적으로 포인터 및 가장 가까운 윈도우 사이의 거리에 해당하는 반경과 스케일링 팩터에 의해 설정되는 반경을 갖는 환형 영역이다. 또한, 스케일링 팩터는 가장 가까운 윈도우에 대한 거리에 좌우되는 변수이므로 비교적 더 크거나 더 작은 영역이 가장 가까운 윈도우의 근접 정도에 따라 분석된다.

<72> 또한, 원이 아닌 다른 기하학적 형상들이 근접 범위에 대해 사용될 수 있다. 예를 들어, 이하에서 좀 더 상세히 기술되는 바와 같이, 근접 범위는 사용자가 주문제작할 수 있는 미리 정해진 방향으로 집중될 수 있다. 일실시예에서, 포인터의 좌측 또는 우측에 집중된 타원은 키스트로크 또는 마우스 버튼 누름과 같은 특정 키 사용자 입력을 사용을 통해 접근 범위를 포인터의 어느 한쪽으로 집중시키는데 사용될 수 있다.

<73> 근접 범위에 사용되는 또 다른 하나의 특히 바람직한 선택 사양 형태는 3차원을 사용하는데, 이 경우에 미리 정해진 범위의 레벨 내의 윈도우만이 근접 포인터에 의해 실행된 것으로 생각될 수 있다. 윈도우는 본 발명에 따른 다수의 방법으로 상이한 레벨에서 구별될 수 있다. 예를 들어, 각 윈도우는 사용자 입력이 현재 수신되고 있는 "활성" 윈도우로 지정되는 상위 레벨 윈도우를 갖는 특정한 레벨로 할당될 수 있다. 근접 범위는 상위 레벨, 또는 다른 임의의 레벨 조합 또는 레벨 범위 상의 윈도우로 제한될 수 있다.

<74> 선택적으로, 다수의 윈도우가 현재 레벨을 공유할 수 있다. 예를 들어, 윈도우의 소정 레벨은 윈도우와 오버랩하는 다른 윈도우의 수에 의해 결정될 수 있다. 또 다른 선택과 같이, 공통 애플리케이션에 의해 소유되는 윈도우는 특정한 레벨 상에 배열되는 특정한 애플리케이션용 윈도우 그룹과 동일한 레벨로 할당될 수 있다.

<75> 상이한 레벨 상의 윈도우를 구별하는 특히 바람직한 하나의 방법은 (윈도우 계층 구조 내의 하위에 윈도우가 위치될 때 레벨수가 증가한다고 가정하면) 선택된 윈도우를 오버랩하는 최하위 레벨 윈도우에 1 레벨을 플러스함으로써 선택된 윈도우의 레벨을 정의한다. 또한, 다른 오버래핑 윈도우를 갖지 않는 임의의 선택된 윈도우는 (예를 들어 0 또는 1의 레벨 수를 갖는) 최상위 레벨 윈도우로서 정의될 수 있다. 또한, 윈도우가 윈도우 계층 구조 내에서 더 위에 배치될 때 레벨 수가 증가함으로써, 선택된 윈도우에 대한 상술한 레벨 결정은 선택된 윈도우를 오버래핑하는 최하위 레벨 윈도우의 레벨로부터 1 레벨을 감산함으로써 일반적으로 동작한다는 것을 이해할 수 있다.

<76> 또 다른 선택에 있어서, 별도의 기하학적 형상 및/또는 반경이 상이한 레벨의 윈도우용으로 사용될 수 있다. 예를 들어, 그것에 의해 근접 범위가 하위 레벨 윈도우의 경우보다 상위 레벨에 위치되는 윈도우에 대한 추가 확장을 가능하게 해준다. 다른 변형을 가하는 것도 통상의 당업자에게 명백히 이해될 수 있다.

<77> 도 6에 도시되는 바와 같이, 루틴(120)은 근접 범위의 원점을 포인터의 현재 위치로 설정함으로써 블록(122)에서 시작한다. 다음에, 블록(124)에서 "탐색 범위(search span)"가 탐색되어 포인터에 가장 가까운 윈도우를 발견한다. 탐색 범위는 통상 원점에 중심을 가지면 미리 정해진 반경으로 확장되는 원이다.

<78> 블록(126)에서는 상기 윈도우가 검색되었는지의 여부에 대한 결정이 실행된다. 검색되지 않았으면, 제어가 블록(128)으로 전달되어 그립 범위를 디폴트 반경으로 설정한다. 그러나, 윈도우가 검색되면, 블록(126)은 제어를 블록(130)으로 전달하여 그립 범위를 동적으로 설정한다. 이것은 원점으로부터 블록(124)에서 검색된 가장 가까운 윈도우 상의 가장 가까운 점까지의 거리를 결정한 다음 결정된 거리에 사용자에게 의해 설정되는 스케일링 팩터를 곱함으로써 수행된다. 이러한 각 단계의 공지된 수학적 연산에

의해 수행될 수 있다.

- <79> 어느 하나의 이벤트에서, 그림 범위의 반경이 설정된 후, 제어는 블록(132)으로 전달된다. 블록(132)은 포인터의 현재 위치에서 정의되는 현재 접근 범위에 대해 실행 윈도우에 적합한 정보를 저장하도록 컴퓨터 시스템에 의해 유지되는 그림 리스트(gripped list)를 삭제한다. 많은 데이터 구조는, 예를 들어, 기록 리스트와 같은 실행 윈도우에 관련되는 각 기록을 갖는 그림 리스트를 구현하는데 사용될 수 있다.
- <80> 그림 리스트가 삭제된 후, 블록(133)이 실행되어 적어도 부분적으로 그림 크기 내에 있는 디스플레이 상의 다음 윈도우를 식별한다. 윈도우의 식별은 예를 들어, 윈도우가 현재 그림 범위에 의해 정의되는 근접 범위를 적어도 부분적으로 오버랩핑한 것으로 검색될 때까지 디스플레이 내의 각 윈도우를 통해 시퀀스함으로써 수행될 수 있다. 이러한 결정은 공지의 윈도우 좌표 및 근접 범위에 기초를 둔 루틴의 수학적 계산이라는 것을 이해하여야 한다. 바람직한 실시예에서, 실행 윈도우는 적어도 부분적으로 근접 범위 내에 배치되는 경계 세그먼트를 갖는 윈도우에 한정된다. 그러나, 또 다른 실시예에서는 완전히 근접 범위를 둘러싸지만 동일한 근접 범위를 교차하는 경계 세그먼트를 갖지 않는 윈도우를 포함하는 것도 역시 바람직할 수 있다. 마찬가지로, 접근 범위에 의해 완전히 둘러싸이는 윈도우를 포함하는 것도 역시 바람직할 수 있다.
- <81> 그 다음에, 결정 블록(134)가 실행되며, 디스플레이 내의 다른 윈도우가 적어도 부분적으로 그림 범위 내에 배치되는 것으로 검색되는 경우, 제어는 블록(136)으로 전달되어 윈도우의 크기를 검색한다. 그 다음, 블록(138)에서 그 윈도우 내의 (예를 들어 교차된 경계 세그먼트와 같은) 실행 경계 세그먼트(affected boundary segment)를 결정한다. 그 정보의 계산이 블록(133)의 실행에서 고유하게 수행될 때 블록(136, 138)이 별도로 필요하지 않을 수 있다.
- <82> 상기 정보를 기초로 하여, 그림 리스트 내의 기록은 블록(139)에서 윈도우용으로 생성된다. 이러한 기록내에 실행 경계 세그먼트가 저장된다. 그 다음에, 제어가 블록(133)으로 복귀하여 그림 범위 내의 다음 윈도우를 식별한다. 디스플레이 내의 각 윈도우는 추가적인 윈도우가 검색되지 않는 시간까지 상술한 방법으로 연속하여 분석된다. 그 다음에, 블록(134)는 루틴(120)을 벗어난다. 그 다음, 그림 리스트는 디스플레이 내의 각 실행 윈도우용 레코드를 포함한다.
- <83> 윈도우 그림 루틴(140)은 도 7에 더욱 상세히 예시된다. 상술한 바와 같이, 루틴(140)은, 예를 들어, 이동 크기 변경 모드 동안 하나 이상의 마우스 버튼을 누르는 것과 같은 동작을 수행함으로써 그림 이벤트를 발생시키는 사용자에 응답하여 초기화된다.
- <84> 루틴(140)은 블록(142)에서 적어도 하나의 윈도우가 그림 리스트 내에 존재하는지의 여부를 결정함으로써 시작한다. 존재하지 않는 경우, 루틴(140)은 현재 근접 범위 내에서 조작하기 위한 윈도우가 존재하지 않기 때문에 즉시 종료된다.
- <85> 그러나, 윈도우가 그림 리스트 내에 존재하는 경우, 제어는 블록(144)으로 전달되어 "그림" 플래그를 설정함으로써 그림 이벤트가 초기화되었다는 것을 표시한다. 그 다음, 블록(146)에서는 포인터가 바람직하게는 이동 또는 변경 포인터 표시 중 어느 한 표시와 구별되는 "그림" 포인터 표시로 설정된다. 대안적으로, 그림 포인터는 임의의 포인터와 동일한 표시를 갖는다. 그러나, 그림 포인터를 구별함으로써, 사용자는 그림 모드가 언제 활성화되는지를 용이하게 식별할 수 있다. 그 다음, 블록(148)에서 실행 윈도우가 상술한 방법 중 임의의 방법으로 강조된다. 그 다음, 디스플레이가 블록(150)에서 갱신됨으로써, 시스템이 그림 모드로 배치되었다는 것을 표시하도록 실행 윈도우 및 포인터를 리프레시한다. 그 후 루틴(140)이 종료된다.
- <86> 포인터 이동 핸들링 루틴(160)이 도 8에 더욱 상세히 예시되어 있다. 통상적으로, 루틴(160)은 사용자가 요구하는 포인터의 이동 거리 및 방향을 정의하는 벡터를 갖고 있다. 벡터는 시작 및 종료 위치에 의해 정의되거나 대안적으로 포인터의 시작 (현재) 위치에 대한 방향 및 거리로 정의될 수 있다. 따라서, 루틴(160)은 그 포인터 벡터를 계산함으로써 블록(162)에서 시작한다.
- <87> 그 다음, 블록(164)에서는 포인터의 위치가 본 기술 분야에 잘 알려진 방법으로 갱신된다.
- <88> 그 후, 블록(166)은 제어를 그림 플래그가 설정되었는지의 여부에 따라 2개의 프로그램 플로우 중 하나로 전환한다. 그림 모드가 설정되지 않았으면, 블록(166)은 제어를 블록(167)으로 전달하여 현재 포인터 모드가 정상 모드인지의 여부를 결정한다. 그렇지 않으면(포인터 모드가 이동 또는 크기 변경 모드 중 어느 한 모드를 표시하지 않으면), 블록(167)은 루틴(120)에 의해 표시되는 실행 윈도우 각각을 강조하는 블록(168 및 170)으로 제어를 전달하기에 앞서 루틴(120)(도 6)을 호출하고 디스플레이를 갱신한다. 현재 포인터 모드가 정상 모드인 경우, 블록(167)은 제어를 블록(170)으로 전달하여 디스플레이 상의 어떤 윈도우도 강조하지 않고 디스플레이를 갱신한다. 어느 하나의 이벤트에서, 블록(170)이 실행된 후, 루틴(160)이 종료된다. 그림 플래그가 설정되지 않고 이동 또는 크기 변경 모드 중 어느 한 모드가 일반적으로 선택되는 경우, 루틴(160)의 프로그램 플로는 포인터가 디스플레이 둘레로 이동함에 따라 그림 리스트 및 실행 윈도우의 강조를 갱신하는 기능을 수행한다.
- <89> 블록(166)으로 복귀하여, 그림 플래그가 설정되는 경우, 제어는 블록(172)으로 전달되어 추가적인 비프로세스형 윈도우(non-processed windows)가 그림 리스트 내에 존재하는지의 여부를 결정한다. 예를 들어, 그림 리스트가 비어 있지 않은 경우, 블록(172)은 처음에 제어를 블록(174)으로 전달하여 그림 리스트로부터 다음 윈도우를 검색한다. 그 다음, 블록(176)이 실행되어 포인터 모드가 이동 또는 크기 변경되는지의 여부를 결정한다. 이동 모드로 설정되면, 제어는 블록(178)으로 전달되어 상기 블록(612)에서 계산된 포인터 벡터를 사용하여 윈도우 위치를 갱신한다. 이러한 방법으로, 윈도우는 포인터와 동시에 이동된다. 그 다음 제어는 블록(172)으로 복귀하여 처리되지 않았던 그림 리스트 내에 추가 윈도우가 존재하는지의 여부를 결정한다.
- <90> 블록(176)으로 복귀하여, 포인터 모드가 이동 모드로 설정되지 않으면, 현재 포인터 모드는 크기 변경 모드인 것으로 가정된다. 따라서, 제어용 블록(180)으로 전달되어 포인터 벡터를 사용하여 윈도우의

크기 및/또는 위치를 갱신한다. 통상적으로, (그 길이가 인접한 세그먼트의 이동으로 인해 변경되었음에도 불구하고) 윈도우는 실행 경계 세그먼트만을 이동시키고 원래 위치에 임의의 비실행 경계 세그먼트를 남김으로써 크기가 변경된다. 그 다음에, 제어는 블록(172)으로 복귀하여 그림 리스트 내의 추가 윈도우를 처리한다.

<91> 일단 그림 리스트 내의 모든 윈도우가 처리되었으면, 블록(172)는 제어를 블록(170)으로 전달하여 디스플레이를 갱신함으로써 실행 윈도우의 위치 및/또는 크기를 리프레시한다. 그 후 루틴(160)이 종료된다.

<92> 각 실행 윈도우가 다수의 방법으로 이동 또는 크기 변경될 수 있다는 것이 명백히 이해될 수 있다. 예를 들어, 포인터 벡터와 방향 및 길이가 동일한 벡터는 윈도우의 이동 및 크기를 변경하는데 사용될 수 있다. 또한, 하나 이상의 실행 경계 세그먼트의 이동을 통해 윈도우의 크기를 변경하는 것은 포인터 벡터의 하나의 성분만을 사용한다. 예를 들어, 주어진 실행 경계 세그먼트의 경우, 실행 경계 세그먼트에 수직인 포인터 벡터의 성분만을 사용하는 것이 바람직할 수 있다. 수직 방향으로 배향된 경계 세그먼트의 경우, 포인터 벡터의 수평 성분만을 사용하는 것이 바람직할 수 있다. 마찬가지로, 수평으로 배향된 경계 세그먼트의 경우, 포인터 벡터의 수직 성분만을 사용하는 것이 바람직할 수 있다. 또한, 윈도우 이동이 본 발명에 따른 수평 및 수직 방향으로 강제로 배향시킬 수 있다는 것도 이해할 수 있을 것이다.

<93> 실행 윈도우의 윤곽(outline)만이 이동 또는 크기가 변경되는 동안 디스플레이될 수 있다는 것도 또한 이해할 수 있을 것이다. 그 결과, 이러한 동작이 이루어지는 동안 다수의 윈도우를 전체적으로 그리고 반복적으로 리드로우(redraw)하는 것과 관련하여 다수의 오버헤드가 추가되는 것을 피하는 것이 가능할 수 있다.

<94> 윈도우 해제 루틴(190)이 도 9에 더욱 상세히 예시된다. 루틴(190)은 루틴(140)에 의해 미리 설정된 그림 플래그를 삭제함으로써 블록(192)에서 시작된다. 다음으로, 블록(194)는 포인터 모드가 현재 이동 모드로 설정되어 있는지의 여부를 결정한다. 이동 모드로 설정되어 있으면, 루틴(100)이 호출되어 도 4에 상술한 바와 같이 이동 모드를 재초기화한다. 이동 모드로 설정되어 있지 않으면, 루틴(110)이 호출되어 도 5에 상술한 방법으로 크기 변경 모드를 재초기화한다. 어느 하나의 이벤트에서, 일단 적절한 모드가 리셋되었으면, 루틴(190)은 종료된다. 따라서, 일단 해제 이벤트가 처리되었으면, 포인터는 그림 이벤트 시점과 동일한 모드로 복귀한다.

<95> 본 발명에 따른 여러 가지 근접 조작 특징을 갖는 동작의 소정 예에 따라, 도 10은 마우스 또는 유사한 장치를 통해 제어되는 포인터(200)과 복수개의 윈도우(210, 220, 230 및 240)이 디스플레이되는 컴퓨터 디스플레이(22)를 예시한다. 윈도우(210)는 4개의 경계 세그먼트(212a-212d)를 구비한 경계(212)를 포함한다. 마찬가지로, 윈도우(220)는 경계 세그먼트(222a-222d)를 갖는 경계(222)를 포함하며, 윈도우(230)는 경계 세그먼트(232a-232d)를 구비한 경계(232)를 포함하며, 윈도우(240)는 경계 세그먼트(242a-242d)를 구비한 경계(242)를 포함한다.

<96> 도 11 내지 도 15는 상술한 방법으로 컴퓨터 시스템에 의해 핸들링되는 다양한 이벤트를 예시한다. 예를 들어, 도 11은 포인터(200)이 **202**로 도시되는 위치에 위치되는 경우 사용자에게 의해 이동 포인터의 모드를 초기화하는 것을 예시한다. 이러한 예에서, 접근 범위의 원점은 **250**으로 도시되며, 탐색 영역이 미리 정해진 반경(252)으로 정의된다.

<97> 상술한 바와 같이, 이동 모드의 초기화가 이루어지면, 루틴(100)이 실행되어 블록(102 및 104) 내에서 포인터 모드를 이동 포인터로 설정한다(도 4). 상기와 같이, 이동 포인터 표시(202)는 이동 포인터를 (도 10의 **200**으로 도시되는) 정상 포인터와 구별하는데 사용될 수 있다.

<98> 다음으로, 루틴(120)이 호출되어 포인터에 대해 정의되는 그림 범위 내의 각 실행 윈도우를 검색한다. 그림 범위의 동적 계산이 도 12에 더욱 상세히 예시되며, 탐색 범위(252) 내에서 가장 가까운 윈도우에 대한 거리가 A로 예시된다. 거리 A를 기초로 하여, 원형 그림 범위(254)는 $A \times X$ 에 해당하는 반경을 갖는 것으로 정의될 수 있는데, X는 사용자에게 의해 설정되는 스케일링 팩터이다.

<99> 도 11로 복귀하여, 윈도우(210)의 경계 세그먼트(212b 및 212c) 및 윈도우(230)의 경계 세그먼트(232d)는 적어도 부분적으로 그림 범위(254) 내에 배치된다는 것을 알 수 있다. 따라서, 상기 윈도우는 실행 윈도우로 결정된다. 그 결과, 이러한 윈도우는, 예를 들어, 아이콘(214 및 224)을 각각 사용하여 강조될 수 있다.

<100> 또한, 도 11은 포인터에 대한 근접 범위의 깊이를 제한하기 위해 그림 범위에 대한 깊이 팩터(depth factor)를 포함하는 것이 가능하다는 것을 예시한다. 이러한 예에서, 임의로 선택된 윈도우의 "레벨"은 선택된 윈도우를 오버래핑하는 최하위 레벨 윈도우의 레벨에 레벨을 추가함으로써 결정되고, 임의의 오버래핑되지 않은(non-overlapping) 윈도우가 "레벨 1"과 같은 상위 레벨로 설정된다. 레벨을 넘버링하는 다른 방법이 선택적으로 사용될 수 있다.

<101> 다른 어떤 윈도우도 윈도우(210 및 220)을 오버래핑하지 않기 때문에, 이러한 윈도우는 본 명세서에서 레벨 0 윈도우로 지정된다. 레벨 0 윈도우에 의해서만 오버래핑되는 윈도우(230)는 레벨 1 윈도우로 지정될 수 있으며, 레벨 0 윈도우(윈도우(220)) 및 레벨 1 윈도우(윈도우(230))에 의해 오버래핑되는 윈도우(240)는 레벨 2 윈도우로 지정될 수 있다. 그림 범위가 레벨 0 및 레벨 1 윈도우만을 포함하도록 설정된다고 가정하면, 윈도우(240)은 그 경계 세그먼트가 그림 범위(254)를 교차한다고 하더라도 실행 윈도우로서 제외될 수 있다. 상술한 바와 같이, 다른 실시예에서 깊이 팩터가 그림 범위 내에 포함될 수 없기 때문에, 윈도우(240)은 실행 윈도우로서 포함될 것이다.

<102> 그림 범위에 대한 깊이 팩터를 실행시키는 또 하나의 특히 바람직한 방법은 기본 레벨을 동적으로 결정하여 기본 레벨로부터 미리 정해진 수의 레벨 내의 윈도우만을 포함하는 것이다. 예를 들어, 그림 범위 원점 또는 반경의 동적 결정과 마찬가지로, 포인터에 가장 가까운 윈도우를 위치시켜 그림 범위에 대한 기본 레벨을 그 가장 가까운 윈도우의 레벨로 설정하는 것이 바람직하다. 그 다음, 그림 범위 내에 속하는 임의의 윈도우의 레벨이 기본 레벨과 비교될 수 있는데, 이 기본 레벨은 가장 가장 가까

운 윈도우에 의해 설정되는 기본 레벨로부터 델타 레벨값 내의 값인 그림 범위 내의 윈도우에 한정된 실행 윈도우를 갖는다. 그림 범위에 대한 깊이 팩터를 실행하는 다른 방법이 본 발명에 따라 사용될 수 있다.

- <103> 도 13에 따르면, 그림 이벤트의 초기화가 예시되어 있다. 상기 예에서, 루틴(140)(도 7)이 실행되어 그림 플래그를 설정하고, 포인터를 그림 포인터로 설정하며, 모든 실행 윈도우를 강조한다. 따라서, 도 13은 원점(250)에 중심을 갖는 그림 포인터 표시(204)의 포인터를 예시한다. 또한, 도 13은 (예를 들어, 윈도우(210, 230) 상에 각각 디스플레이되는 별도의 아이콘(216, 236)을 사용하여) 별도의 강조가 수행되어 그리핑(gripping)하기에 앞서 실행 윈도우와 그림된 윈도우를 구별할 수 있는 특징을 예시한다.
- <104> 그림 이벤트가 추가화된 후, 포인터를 이동시키면 포인터의 현재 모드에 따라 실행 윈도우의 이동 또는 실행 윈도우의 크기 변경 중 어느 하나가 수행된다. 예를 들어, 도 14에 도시되는 바와 같이, 포인터가 이동 모드에 있는 경우, 포인터 벡터(208)을 따라 위치(204)로부터 위치(206)으로 포인터를 이동시키면 경계(212')의 각 세그먼트가 벡터(218)을 따라 이동되었던 210'로 도시되는 위치로 벡터(218)을 따라 윈도우(210)의 이동이 일어난다. 또한, 윈도우(230)은 벡터(208 및 218)와 방향 및 거리가 동일한 벡터(238)을 따라 이동하여 (새로운 경계부(232')와 함께) 도면번호(230')로 식별되는 위치로 새로 위치된다.
- <105> 또한, 도 14로부터 명백히 이해될 수 있는 바와 같이, 포인터의 이동동안 수행되는 추가적인 단계는 실행 윈도우를 활성화하거나 또는 적어도 윈도우 각각을 상위 레벨로 올리는 단계이다. 예를 들어, 윈도우(230)은 현재 윈도우(220)을 오버래핑하는 레벨 0 윈도우로 올라가는 것으로 도시된다.
- <106> 도 15는 동일한 벡터(208)을 따라 위치(204)로부터 위치(206)로 포인터의 이동을 예시하지만, 포인터가 크기 변경 모드 상태에 있지는 않다. 이러한 예에서, 각 윈도우(210, 230)의 실행 윈도우 세그먼트는 상기 세그먼트에 수직인 포인터 벡터의 성분을 따라 이동한다.
- <107> 예를 들어, 윈도우(210)의 경계 세그먼트(212b)는 위치(212b')까지 벡터(218a)로 따라 이동된다. 이러한 이동은 포인터 벡터(208)의 수평 성분에 해당하는 수평 방향 및 거리를 갖는 벡터(218a)를 따라 일어난다. 마찬가지로, 윈도우(210)의 경계 세그먼트(212c)는 위치(212')까지 벡터(218b)를 따라 이동된다. 벡터(218b)는 벡터(208)의 수직 성분에 해당하는 수직 방향 및 거리를 갖는다.
- <108> 윈도우(230)의 경우, 경계 세그먼트(232d)만이 실행됨으로써 이러한 세그먼트가 (벡터(218a) 및 벡터(208)의 수평 성분과 일치하는 수평 방향 및 길이를 갖는) 벡터(238a)를 따라 이동한다. 또한, 윈도우(230)는 크기 변경 모드에서 포인터의 이동에 응답하여 레벨 0으로 활성화되거나 레벨 0으로 올라가는 것으로서 예시된다. 그러나, 다른 실시예에서는 그림 이벤트에 응답하여 윈도우의 현재 레벨의 변경이 수행되지 않을 수도 있다는 것이 명백히 이해될 수 있다.
- <109> 다양한 변경이 본 발명의 정신과 범위를 벗어남이 없이 본 명세서에 기술되는 실시예에 대해 이루어질 수 있다. 예를 들어, 단지 크기 변경 모드 또는 이동 모드만이 컴퓨터 시스템에 사용될 수 있다는 것이 명백히 이해될 수 있다.
- <110> 또한, 임의의 크기 변경, 이동, 또는 그림 모드를 초기화하는 다른 방법이 본 발명에 따라 사용될 수 있다는 것이 명백히 이해될 수 있다. 예를 들어, 별도의 이동 및 크기 변경 그림 모드를 사용하여 어느 하나의 그림 모드를 초기화하기 전에 실행 윈도우의 임의의 강조를 제거하는 것이 바람직할 수 있다. 그것에 의해, 그림 이벤트를 초기화하기에 앞서 특정한 이동 또는 크기 변경 모드로 포인터를 배치시키는 것을 사용자에게 요구하지 않는다. 대신에, 별도의 사용자 입력 조합이 각 그림 모드에 대해 사용된다. 일 실시예로서, 사용자는 키보드 상의 제어 키를 누르면서 마우스 버튼을 눌러 이동 그림 모드로 들어가거나 쉬프트 키를 누르면서 마우스 버튼을 눌러 크기 변경 그림 모드로 들어갈 수 있다.
- <111> 다른 실시예에서, 단속적 "프리뷰"(intermittent preview) 모드를 사용함으로써, 포인터가 미리 정해진 위치에 있는 동안 사용자는 키 및/또는 마우스 버튼을 누를 수 있으므로 특정한 이동 또는 크기 변경 모드로 포인터를 다시 배치시키지 않고도 사용자로 하여금 어느 윈도우가 실행되었는지를 간단히 알 수 있게 해주며, 그 후 사용자가 이동 그림 모드 또는 크기 변경 모드를 선택하여 원하는 동작을 수행할 수 있게 하는 것이 바람직하다. 다른 실시예에서, 대응하는 포인터의 이동없이 이동 그림 모드 또는 크기 변경 그림 모드를 초기화하면 사용자가 실행 윈도우의 현재 그룹을 이동시키거나 또는 크기를 변경시키는 것이 바람직한지의 여부를 결정할 수 있도록 "프리뷰" 기능이 충분히 제공될 수 있다.
- <112> 도 16은 본 발명에 따라 사용될 수 있는 또 다른 특징, 즉, 주로 특정한 사용자 입력에 응답하여 미리 정해진 방향을 따라 연장되는 집중식 그림 범위를 사용하는 것을 예시한다. 이러한 도면에 예시되는 바와 같이, 컴퓨터 디스플레이(22')는 원점(272)을 정의하는 포인터(170)를 디스플레이한다. 집중식 그림 범위(274)는 주로 포인터(270)의 우측으로 연장되는 타원으로 정의된다. 마찬가지로, 포인터(270)에 근접하지만 포인터의 왼쪽에 위치되는 윈도우(278)은 실행되지 않지만, 윈도우(280 및 290)은 그림 범위가 경계 세그먼트(282b 및 282c)와 경계(292)의 세그먼트(292a) 및 (292b)와 교차하기 때문에 실행된다. 따라서, 윈도우(280 및 290)는 아이콘(284 및 294)를 통해 각각 실행 윈도우로서 지정되어 도시된다.
- <113> 상술한 바와 같이, 원형 그림 범위에 대한 선택적으로 집중식 그림 범위를 사용하는 것은, 예를 들어, 특정 키스트로크 및/또는 마우스 누름과 같은 특정한 사용자 입력을 통해 수행될 수 있다. 예를 들어, 이것은 원하는 경우 사용가능한 형태중 사용자가 선택한 형태를 갖는 다수의 선택적인 그림 범위 형태의 사용을 가능하게 해줄 수 있다. 또한, 하나의 그림 범위의 선택이 그림 이벤트를 초기화하기에 앞서 수행되어 사용자는 특정한 그림 범위에 대한 실행 윈도우를 미리 볼 수 있다.
- <114> 하나의 가능한 형태로서, 사용자는 좌우측 마우스 버튼을 상이하게 누름으로써 좌우측 집중식 그림 범위 및 원형 그림 범위 사이를 선택할 수 있다. 단독으로 좌측 마우스 버튼을 누르면 그림 범위가 좌측에 집중되는 반면, 단독으로 우측 마우스 버튼을 누르면 그림 범위가 우측에 집중된다. 좌우측 마우스 버튼 양자를 동시에 누르면 원형 그림 범위가 된다. 기타 다른 변경이 이루어질 수 있다는 점을 통상의 당업자는 명백히 이해할 것이다.

<115> 윈도우 등의 동시 발생하는 깊이 조작

<116> 다수의 추가적인 변경이 본 발명에 따라 이루어질 수 있다. 예를 들어, 도 17은 "변경 깊이" 이벤트를 검출함으로써 제어를 도 18에 예시되는 깊이 변경 루틴(300)으로 전달하는 루틴(50) 내의 추가적인 블록(70)을 예시한다. 특히, 윈도우가 근접 포인터에 의해, 예를 들어, 사용자에게 의해 제공되는 추가적인 사용자 입력에 응답하여 그리핑(gripped)되는 경우 윈도우의 깊이 (또는 레벨)을 선택적으로 변경하는 것이 바람직할 수 있다. 이러한 특징은 크기 변경 및 이동 모드 양자에서 동작가능할 수 있으며, 좀더 바람직하게는 이동 모드에서만 동작 가능할 수 있다. 상기 이벤트를 초기화하는 특정한 사용자 입력은 많은 사용자 입력 조합을 토대로 한다. 그런데, 예를 들어, 이동 모드에서는 키보드 상의 상하 화살표의 누름에 응답하여 깊이 변경 이벤트(change depth event)를 초기화하는 것이 바람직할 수 있다. 이러한 방법으로, 사용자는 컴퓨터 디스플레이 상에서 디스플레이되는 윈도우의 계층 구조 내에서 실행 윈도우의 그룹을 상하로 이동시키는 것이 가능할 수 있다. 따라서, 컴퓨터 시스템 상에 디스플레이되는 상이한 윈도우에 대해 동작할 수 있다는 점에서 더 큰 융통성이 제공될 수 있다.

<117> 도 18에 제공되는 바와 같이, 루틴(300)은 그림 플래그가 설정되었는지의 여부를 결정함으로써 블록(302)에서 시작한다. 그림 플래그가 설정되지 않았으면, 루틴(300)이 종료되고 깊이 변경 이벤트가 무시된다. 대안적으로, 그림 플래그를 설정하기 전에 앞서 윈도우 깊이를 조작하도록 하는 것이 바람직할 수 있다. 그림 플래그가 설정되어 있는 경우, 블록(302)은 제어를 블록(304)으로 전달하여 그림 리스트 내에서 각 윈도우를 처리하는 루프를 초기화한다. 윈도우가 그림 리스트에 저장되지 않는 경우, 제어는 즉시 블록(306)으로 전달되어 디스플레이 및 종료 루틴을 갱신한다. 그러나, 추가적인 비처리 윈도우가 그림 리스트에 있는 한, 제어가 블록(308)으로 전달되어 그림 리스트로부터 다음 윈도우를 검색한 다음 블록(310)이 실행되어 깊이 변경 이벤트는 실행 윈도우를 또한 윈도우의 "상향이동(promoting)" 또는 "하향이동(demoting)"으로 불리우는 윈도우 계층(hierarchy) 내에서 상방향 이동시키기 위한 요구인지 또는 하방향으로 이동시키기 위한 요구인지의 여부를 결정한다.

<118> (예를 들어, "상방향" 화살표 키의 누름에 응답하여) 실행 윈도우를 상방향으로 이동시키기 위한 요구인 경우, 제어는 블록(312)으로 전달되어 윈도우가 디스플레이 상에서 모든 비실행 윈도우 상에 있는 레벨로 되어 있는지의 여부를 결정한다. 윈도우가 디스플레이 상에서 모든 비실행 윈도우 상에 있는 레벨로 되어 있는 경우, 다른 실행 윈도우의 상대 레벨이 유지된다고 가정하면 윈도우는 가능한 상위 레벨로 용이하게 배치된다. 따라서, 제어는 블록(304)으로 전환되어 현재 윈도우의 레벨을 변경시킴 없이 다음 윈도우를 처리한다.

<119> 윈도우가 모든 비실행 윈도우 상에 배치되지 않는 경우, 추가된 윈도우를 처리하기 위해 제어가 블록(304)으로 복귀하기 전에 제어는 블록(314)으로 전달되어 윈도우를 1 레벨만큼 상방향으로 이동시킨다. 윈도우를 상향이동시키는 것은 소정 환경에서 디스플레이 상의 다른 윈도우의 하향이동(demotion)을 요구할 수 있다.

<120> 블록(310)으로 돌아가서, 하향 이동 요구가 (예를 들어, "하향" 화살표 키의 누름에 응답하여) 초기화되었으면, 블록(316)이 실행되어 윈도우가 모든 비실행 윈도우의 레벨 이하인 레벨에 있는지를 결정한다. 윈도우가 모든 비실행 윈도우의 레벨 이하인 레벨에 있는 경우, 제어는 윈도우의 현재 윈도우의 레벨을 변경시킴없이 블록(304)으로 전환된다. 그러나, 윈도우가 모든 비실행 윈도우 아래에 배치되지 않는 경우, 제어는 블록(318)으로 전달되어 제어가 블록(304)으로 복귀하기 전에 윈도우를 하향이동시킨다.

<121> 일단 그림 리스트 내의 모든 윈도우가 처리되었으면, 제어는 블록(304)에 의해 블록(306)으로 전달되어 디스플레이를 갱신하고 그 후에 루틴을 종료시킨다.

<122> 통상적으로, 윈도우는 상향 이동 요구 또는 하향 이동 요구에 응답하여 1 레벨만큼 상향이동 또는 하향이동된다. 대안적으로, (예를 들어, 좌우 방향으로 또는 상하 방향으로) 실행 윈도우를 동적이며 선택적으로 상향이동 또는 하향이동시키는데 포인터의 이동을 사용하는 것이 바람직하다. 다른 변경이 본 기술 분야의 당업자에게 명백히 이해될 수 있을 것이다.

<123> 윈도우 등에서 동시 발생하는 확장/ 압축

<124> 도 17로 돌아가서, 본 명세서에 기술되는 실시예에 의해 구현될 수 있는 또 다른 특징은 확장/압축 특징이다. 이러한 특징은 루틴(50)의 블록(72)에서 검출되며 루틴(330)에 의해 핸들링되는 이벤트에 의해 초기화된다. 확장/압축 특징은 일반적으로 공통 원점에 대해 내부 또는 외부 중 어느 한 방향으로 윈도우를 이동 및/또는 크기 변경시킴으로써 동작한다. 일반적으로, 상기 윈도우의 이동 또는 크기 변경은 포인터의 현재 위치에 대해 방사상으로 수행될 수 있다. 그러나 선택적인 실시예에서, 원점에 대해 일반적으로 내부 또는 외부로 향하는 다른 방향이 본 발명에 따라 정의될 수 있다.

<125> 통상적으로, 확장/압축 이벤트는 그림 이벤트가 활성화된 후 초기화되어 확장/압축 이벤트에 의해 조작되는 윈도우는 미리 정해진 윈도우로 제한된다. 선택적으로, 확장/압축 이벤트는 그림 모드로 들어가기 전에 사용될 수 있다. 또한, 확장/압축 특징이 본 발명에 따른 다른 접근 특징과 별도로 제공될 수 있다.

<126> 많은 사용자 입력 조합이 확장/압축 이벤트를 초기화하는데 사용될 수 있다. 예를 들어, "좌방향" 화살표 키 누름 동작이 압축 요구를 초기화하는데 사용될 수 있는 반면, "우방향" 화살표 키 누름 동작은 확장 요구를 초기화하는데 사용될 수 있다. 그러나, 다른 사용자 입력 조합이 대안적으로 사용될 수 있다.

<127> 도 19에 도시되는 바와 같이, 예를 들어, 루틴(330)은 우선 블록(332)에서 그림 플래그가 설정되었는지의 여부를 검출함으로써 실행된다. 플래그가 설정되지 않았으면, 루틴(330)은 종료되어 이벤트를 무시한다. 그러나, 그림 플래그가 설정되었으면, 제어는 블록(334)으로 전달되어 그림 리스트 내의 각 윈도우를 처리하도록 루프를 초기화한다. 블록(338)에서, 그림 리스트로부터 차기 윈도우가 검색된다. 다음으로, 블록(340)에서 원점(통상 포인터에 의해 지시되는 현재 위치)으로부터 윈도우 상의 가장 가까운 점으로 연장되는 방사상 라인이 계산된다. 그 다음에, 이벤트가 실행 윈도우를 확장하기 위한 요구인지 또

는 압축하기 위한 요구인지의 여부에 따라, 블록(342)는 제어를 블록(344 또는 350) 중의 어느 한 블록으로 전달한다.

- <128> 이벤트가 실행 윈도우를 확장하기 위한 (예를 들어, 원점으로부터 외부 방향으로 각 윈도우를 이동시키거나 크기를 변경하기 위한) 요구를 포함하는 경우, 블록(344)가 실행된다. 포인터 모드가 이동 모드로 설정되는지 크기 변경 모드로 설정되는지의 여부에 따라, 제어는 블록(346 또는 348) 중 어느 한 블록으로 전달된다. 포인터 모드가 이동 모드로 설정되는 경우, 블록(346)은 윈도우에 대해 정의되는 방사상 위치를 따라 외부 방향으로 윈도우 위치를 갱신한다. 통상적으로, 소정의 애플리케이션에서 미리 정해진 증가량이 포인터의 이동, 마우스 버튼이 눌러지는 시간의 양 등과 일치하는 것이 바람직하지만, 미리 정해진 증가량이 윈도우 위치를 갱신하는데 사용된다. 이러한 전자의 예로는 예를 들어 사용자로 하여금 실행 윈도우를 연속적인 범위 내에서 동적으로 확장 또는 압축을 가능하게 해준다.
- <129> 블록(344)로 돌아가서, 포인터 모드가 크기 변경 모드로 설정되는 경우, 블록(348)이 실행되어 방사상 라인을 따라 외부 방향으로 윈도우 크기 및/또는 위치를 갱신하여, 통상적으로 윈도우 크기가 감소된다. 통상적으로, 비실행 세그먼트를 고정 위치로 유지하는 동안, 이러한 방법으로 윈도우의 크기를 변경하는 것은 윈도우의 실행 경계 세그먼트를 이동시킴으로써 수행된다. 윈도우의 크기 변경은 방사상 라인의 방향을 따라 수행되거나 그 성분(예를 들어, 임의의 주어진 실행 경계 세그먼트에 수직인 방향의 성분)만을 사용할 수 있다.
- <130> 블록(342)로 돌아가서, 압축 요구가 이벤트 내에 통합되는 경우, 블록(350)이 실행되고 포인터 모드가 이동 모드로 설정되는지 크기 변경 모드로 설정되는지의 여부에 따라 블록(352 또는 354) 중 한 블록이 실행된다.
- <131> 포인터가 이동 모드로 설정되는 경우, 블록(352)는 통상 상술한 바와 같이 고정된 증분을 사용하여 윈도우 위치를 방사상 방향을 따라 내부 방향으로 갱신한다. 그러나, 포인터 모드가 크기 변경 모드로 설정되는 경우, 윈도우 크기 및/또는 위치는 블록(354)에서 블록(348)과 매우 동일한 방법으로 방사상 라인을 따라 내부 방향으로 갱신된다. 임의의 블록(346, 348, 352 또는 354)가 실행된 후, 모든 윈도우의 처리가 이루어지는 시간까지 제어는 블록(334)으로 복귀하여 그림 리스트 내의 각 추가적인 윈도우를 처리한다. 그 다음에, 블록(336)이 실행되어 디스플레이를 갱신하고 루틴(330)을 종료시킨다.
- <132> 본 발명에 따른 확장/압축 특성의 동작의 예로서, 도 20은 포인터(401)이 디스플레이되는 디스플레이(400)과 추가적인 윈도우(450)를 오버래핑하는 4개의 윈도우(410, 420, 430 및 440)를 예시한다. 포인터(401)은 원점(402 및 404)으로 정의되는 그림 범위를 갖는 것으로 도시된다. 도 20은 그림 이벤트보다 먼저 컴퓨터 시스템의 상태를 예시한다. 윈도우(410) 내지 윈도우(440) 각각이 각각의 경계부(412, 422, 432 및 442)로부터의 실행 경계 세그먼트를 포함하기 때문에, 윈도우(410 내지 440) 각각은 아이콘(414, 424, 434 및 444)을 통해 각각 실행 윈도우가 지정된다.
- <133> 도 21은 확장 요구의 초기화가 이루어진 후 뿐만 아니라 그림 이벤트가 초기화된 후에 루틴(330)에 의해 상술한 방법으로 핸들링되는 디스플레이(400)를 예시한다. 각 윈도우(410 내지 440)의 경우, 방사상 라인(415, 425, 435 및 445)은 각각 원점(402)로부터 윈도우 상의 가장 가까운 점까지 계산된다. 그 다음에, 각 윈도우는 **410', 420', 430' 및 440'**로 각각 예시되는 위치로 미리 정해진 증분만큼 외부 방향으로 이동된다.
- <134> 압축 이벤트는 동일한 방법으로 동작하여 미리 정해진 증분만큼 각 윈도우를 원점(402)에 더 가까이 이동시킨다는 것을 명백히 이해할 수 있을 것이다. 또한 한계가 확장 및 압축 함수 중 중 어느 하나로 정해져 윈도우는 원점(402)에 대해 동일한 방향 뿐만 아니라 디스플레이 내에도 유지된다는 것도 명백히 이해할 수 있을 것이다. 대안적으로, 원점(402)를 초과하여 내부 방향으로 이동하면, 윈도우는 원점에 대해 거울 대칭이 된다.
- <135> 또한, 실행 윈도우 세그먼트만이 이벤트에 의해 변경된다 하더라도, 윈도우가 이동함에 따라 매우 동일한 방법으로 내부 또는 외부 방향으로 크기 변경이 발생한다는 것을 명백히 이해할 수 있을 것이다. 다른 변경이 통상의 당업자에게 명백히 이해될 수 있을 것이다.
- <136> 특정 컴퓨터 소프트웨어 애플리케이션 뿐만 아니라 운영 체제의 경우와 같은 그래픽 사용자 인터페이스 환경의 접근 조작 특징을 갖는 무수한 애플리케이션 중에서, 상기 특징 중 소정의 특정 용도가 추상 스택 애플리케이션(abstraction stack application)에 사용된다. 일반적으로, 추상 스택은 통상 특정한 관심 주제에 관한 정보의 수집을 나타내는 지체(body of knowledge: BOK)로부터의 정보를 표시한다. 추상 스택은 컴퓨터 디스플레이 상의 추상의 상이한 레벨과 연관된 다수의 정보 요소를 동시에 디스플레이하고 3차원 작업 공간에서 컴퓨터 디스플레이 상의 정보 요소를 가시적으로 링크시켜 추상 레벨의 계층 배열을 표시한다.
- <137> 예를 들어, 도 22는 깊이 벡터(502)를 따라 정의되는 복수개의 렌더형 객체(rendered object)를 포함하는 추상 스택(500)을 예시한다. 한 쌍의 컴퍼스 핸들(compass handle)(510 및 512)는 자신들 사이에 삽입되는 교차점(514a) 및 렌즈(570, 572, 574)와 같은 중간 객체를 갖는 추상 스택의 단부를 정의한다. 교차점(514a)은 지체용 추상 레벨을 나타내며, 각 렌즈는 지체 내의 하나 이상의 추상 레벨로부터 정보를 디스플레이하는 특정 필터 구성으로 정의된다. 결합 밴드(505)는 깊이 벡터를 따라 배치되어 3차원 작업 공간에서 객체의 계층 관계를 가시적으로 표시하는 것을 돕는 객체들 사이에서 연장된다.
- <138> 추상 객체에서, 예를 들어, 렌즈(574)를 오버래핑하는 점선으로 표시된 렌즈(570, 572)로 도시되는 바와 같이, 소정 렌즈가 다른 렌즈를 오버래핑함으로써 오버래핑된 렌즈에 제공되는 정보의 일부를 가리는 것이 가능하다. 그러나, (예를 들어, 렌즈(574)가 그 사용자 선택을 통해 활성화되며 디스플레이의 상위층으로 될 수 있는 종래의 윈도우와 일치하면) 렌더형 객체의 계층 관계가 유지되도록 렌즈(574)가 렌즈(570 또는 572) 중 어느 한 렌즈를 오버래핑하지 못하게 하는 것이 바람직하다. 따라서, 본 발명에 따르면, 접근 조작 모드에서 포인터(576)을 예를 들어 벡터(578)를 따라 **576'**으로 표시된 지점으로 이동시키는 경우 (근접 범위(577) 내에 있는) 렌즈(570, 572)를 각각 **570'**과 **572'**으로 나타낸 지점으로 이동시킬 수 있어서 렌즈(574)의 숨겨진 정보를 드러내게 한다.

<139> 그러나, 본 발명이 추상 스택 애플리케이션과 관련하여 단 하나의 용도로 제한되지 않는다는 것을 명백히 이해할 수 있을 것이다.

<140> 본 발명의 본질과 범위를 벗어남이 없이 본 명세서에 기술된 실시예를 다양하게 변경시킬 수 있다는 점을 이해할 수 있다. 따라서, 본 발명은 본 명세서에 첨부된 특허 청구 범위에 의해서만 정해진다.

발명의 효과

<141> 본 발명의 윈도우 등은 포인터의 이동에 응답하여 동시에 이동되거나 크기가 변할 수 있다. 또한, 본 발명의 윈도우 등은 포인터 가까이에 위치되는 공통 원점(common origin)에 대해 안쪽 또는 바깥쪽 중 어느 하나로 동시에 이동되거나 크기가 변할 수 있다. 따라서, 사용자는 좀 더 조직적인 방식으로 다중 구성 요소를 더욱 신속하고 용이하게 이동시키며 크기가 변하도록 해준다.

(57) 청구의 범위

청구항 1

컴퓨터 디스플레이 상에서 복수개의 윈도우를 조작하는 방법에 있어서,

- a) 다수개의 윈도우를 컴퓨터 디스플레이 상에 디스플레이하는 단계;
- b) 포인터를 컴퓨터 디스플레이 상에 디스플레이하는 단계;
- c) 다수개의 윈도우로부터 제1 및 제2 윈도우를 실행 윈도우들-상기 실행 윈도우는 적어도 부분적으로는 포인터에 인접한 원점 둘레에 정의되는 근접 범위 내에 배치되는 것임-로서 정의하는 단계; 및
- d) 포인터를 이동시키기 위한 사용자 입력에 응답하여, 실행 윈도우를 동시에 조작하는 단계를 포함하는 윈도우 조작 방법.

청구항 2

제1항에 있어서, 각 윈도우는 적어도 하나의 경계 세그먼트를 구비한 경계부를 포함하고, 각 실행 윈도우는 근접 범위 내에 적어도 부분적으로 배치되는 실행 경계 세그먼트를 포함하며, 상기 실행 윈도우를 동시에 조작하는 단계는 적어도 실행 윈도우의 실행 경계 세그먼트를 동시에 이동시키는 단계를 포함하는 윈도우 조작 방법.

청구항 3

제2항에 있어서, 상기 적어도 실행 윈도우들의 실행 경계 세그먼트들을 동시에 이동시키는 단계는 실행 윈도우들을 이동시키는 단계를 포함하는 윈도우 조작 방법.

청구항 4

제2항에 있어서, 상기 실행 윈도우의 실행 경계 세그먼트를 적어도 동시에 이동시키는 단계는 각 실행 윈도우의 경계 내에 적어도 하나의 추가적인 경계 세그먼트를 고정 위치로 유지시킴으로써 실행 윈도우의 크기를 변경시키는 단계를 포함하는 윈도우 조작 방법.

청구항 5

제2항에 있어서, 제1 벡터를 따라 포인터를 이동시키기 위한 사용자 입력을 수신하는 단계를 추가로 포함하며,

상기 실행 경계 세그먼트들을 동시에 이동시키는 단계는 제2 벡터를 따라 적어도 하나의 실행 윈도우의 실행 경계 세그먼트를 이동시키는 단계를 포함하는

윈도우 조작 방법.

청구항 6

제5항에 있어서, 상기 제2 벡터가 제1 벡터와 동일한 방향으로 연장되는 윈도우 조작 방법.

청구항 7

제5항에 있어서, 상기 제2 벡터가 실행 경계 세그먼트에 수직한 방향을 갖는 윈도우 조작 방법.

청구항 8

제2항에 있어서, 사용자 입력에 응답하여 원점에 대해 내부 및 외부 방향 중 한 방향으로 적어도 실행 윈도우의 실행 경계 세그먼트의 각각을 동시에 이동시키는 단계를 추가로 포함하는 윈도우 조작 방법.

청구항 9

제1항에 있어서, 상기 제1 및 제2 윈도우를 소정의 사용자 입력에 응답하여 실행 윈도우로서 정의하는 것을 선택적으로 인에이블링하는 단계를 추가로 포함하는 윈도우 조작 방법.

청구항 10

제9항에 있어서, 상기 소정의 사용자 입력이 소정의 사용자 선택 모드를 포함하는 윈도우 조작

방법.

청구항 11

제10항에 있어서, 상기 소정의 사용자 입력은 이동 모드 및 크기 변경 모드 중 하나의 사용자 선택 모드를 포함하고, 상기 이동 모드에서는 상기 실행 윈도우들을 동시에 조작하는 단계가 실행 윈도우들을 이동시키는 단계를 포함하며, 상기 크기 변경 모드에서는 상기 실행 윈도우를 동시에 조작하는 단계가 실행 윈도우의 크기를 변경하는 단계를 포함하는 윈도우 조작 방법.

청구항 12

제9항에 있어서, 상기 포인터를 디스플레이하는 단계가 제1 및 제2 윈도우를 실행 윈도우로 정의하는 것이 인에이블되는지의 여부에 응답하여 제1 및 제2 포인터 표시 중 하나의 표시를 선택적으로 디스플레이하는 단계를 포함하는 윈도우 조작 방법.

청구항 13

제9항에 있어서, 상기 실행 경계 세그먼트들을 적어도 동시에 이동시키는 단계는 소정의 제2의 사용자 입력에 응답하여 수행되는 윈도우 조작 방법.

청구항 14

제13항에 있어서, 상기 소정의 제2의 사용자 입력은 제1 및 제2 윈도우가 실행 윈도우로서 인에이블되는 동안 적어도 하나의 마우스 버튼의 누름을 포함하는 윈도우 조작 방법.

청구항 15

제13항에 있어서, 컴퓨터 디스플레이 상에서 실행 윈도우를 강조하는 단계를 추가로 포함하는 윈도우 조작 방법.

청구항 16

제15항에 있어서, 상기 실행 윈도우를 강조하는 단계는 소정의 제2의 사용자 입력을 수신하기에 앞서 상기 실행 윈도우를 강조하는 단계를 포함하는 윈도우 조작 방법.

청구항 17

제1항에 있어서, 상기 원점을 포인터에 의해 지시되는 위치로 정의하는 단계를 추가로 포함하는 윈도우 조작 방법.

청구항 18

제1항에 있어서, 상기 근접 범위는 원점 둘레를 둘러싸는 타원을 포함하는 윈도우 조작 방법.

청구항 19

제18항에 있어서, 상기 근접 범위는 원점을 중심으로 형성되는 원을 포함하는 윈도우 조작 방법.

청구항 20

제18항에 있어서, 사용자 입력에 응답하여 근접 범위로서 제1 및 제2 기하학적 형성 사이에서 선택하는 단계를 추가로 포함하며,

상기 제1 및 제2 기하학적 범위는 각각이 원점을 둘러싸며 통상 서로의 범위로부터 대향 연장되는

윈도우 조작 방법.

청구항 21

제1항에 있어서, 상기 다수개의 윈도우는 다수개의 레벨중 어느 한 레벨로 각각 정의되고,

사용자 입력에 응답하여 각 실행 윈도우의 레벨을 선택적으로 증가 또는 감소시키는 단계를 추가로 포함하는

윈도우 조작 방법.

청구항 22

제1항에 있어서, 상기 다수개의 윈도우는 다수개의 레벨중 어느 한 레벨에서 각각 정의되고, 상기 근접 범위는 소정 범위의 레벨과 관련된 깊이를 포함하며, 상기 제1 및 제2 윈도우를 실행 윈도우로 정의하는 단계는 소정 범위의 레벨로 정의된 윈도우들만을 실행 윈도우로 정의하는 단계를 포함하는 윈도우 조작 방법.

청구항 23

제22항에 있어서, 상기 근접 범위는 소정 범위의 레벨로 정의되는 별도의 기하학적 형상을 포함하는 윈도우 조작 방법.

청구항 24

제 1항에 있어서, 상기 실행 윈도우들을 동시에 조작하는 단계가 제1 윈도우를 활성화하는 단계

를 포함하는 윈도우 조작 방법.

청구항 25

제1항에 있어서, 상기 실행 윈도우들을 동시에 조작하는 단계는 실행 윈도우들의 위치를 갱신하는 단계를 포함하고,

상기 컴퓨터 디스플레이를 리프레시(refresh)하는 단계를 추가로 포함하는 윈도우 조작 방법.

청구항 26

제25항에 있어서, 상기 컴퓨터 디스플레이를 리프레시하기에 앞서 포인터의 위치를 갱신하는 단계를 추가로 포함하는 윈도우 조작 방법.

청구항 27

제1항에 있어서, 상기 실행 윈도우들을 동시에 조작하는 단계는, 실행 윈도우들을 닫는 단계, 실행 윈도우들을 개별 아이콘으로 최소화하는 단계, 실행 윈도우들을 단일 아이콘으로 최소화하는 단계, 및 이들의 결합 단계로 구성되는 군으로부터 선택되는 실행 윈도우 상에서 동작을 수행하는 단계를 포함하는 윈도우 조작 방법.

청구항 28

컴퓨터 시스템에 있어서,

a) 다수개의 윈도우 및 포인터가 디스플레이되는 컴퓨터 디스플레이;

b) 사용자 입력을 수신하도록 구성되는 사용자 입력 장치; 및

c) 컴퓨터 디스플레이 및 사용자 입력 장치에 결합되고, 다수개의 윈도우로부터 제1 및 제2 윈도우를 실행 윈도우들 - 상기 실행 윈도우는 적어도 부분적으로는 포인터에 인접한 원점 둘레에 정의되는 근접 범위 내에 배치되는 것임 - 로서 정의하며, 포인터를 이동시키는 사용자 입력에 응답하여 상기 실행 윈도우들을 동시에 조작하도록 구성되는 프로세서

를 포함하는 컴퓨터 시스템.

청구항 29

i) 다수개의 윈도우를 컴퓨터 디스플레이 상에 디스플레이하는 단계;

ii) 포인터를 컴퓨터 디스플레이 상에 디스플레이하는 단계;

iii) 다수개의 윈도우로부터의 제1 및 제2 윈도우들을 실행 윈도우-상기 실행 윈도우는 적어도 부분적으로는 포인터에 인접한 원점 둘레에 정의되는 근접 범위 내에 배치되는 것임-로서 정의하는 단계 및

iv) 포인터를 이동시키기 위한 사용자 입력에 응답하여, 상기 실행 윈도우들을 동시에 조작하는 단계

를 포함하여 컴퓨터 디스플레이 상에서 다수개의 윈도우를 조작하는 방법을 컴퓨터에서 수행하도록 구성되는 컴퓨터 프로그램을 포함한 컴퓨터 판독가능한 기록 매체.

청구항 30

컴퓨터 디스플레이 상에서 다수개의 그래픽 사용자 인터페이스 구성 요소를 조작하는 방법에 있어서,

a) 다수개의 그래픽 사용자 인터페이스 구성 요소를 컴퓨터 디스플레이 상에 디스플레이하는 단계;

b) 포인터를 컴퓨터 디스플레이 상에 디스플레이하는 단계;

c) 다수개의 그래픽 사용자 인터페이스 구성 요소로부터의 제1 및 제2 그래픽 사용자 인터페이스 구성 요소들을 실행 그래픽 사용자 인터페이스 구성 요소 -상기 실행 그래픽 사용자 인터페이스 구성 요소는 적어도 부분적으로는 포인터에 인접한 원점 둘레에 정의되는 근접 범위 내에 배치되는 것임-로서 정의하는 단계; 및

d) 포인터를 이동시키기 위한 사용자 입력에 응답하여 상기 실행 그래픽 사용자 인터페이스 구성 요소들을 동시에 조작하는 단계

를 포함하는 그래픽 사용자 인터페이스 구성 요소 조작 방법.

청구항 31

제30항에 있어서, 상기 다수개의 그래픽 사용자 인터페이스 구성 요소가 적어도 하나의 윈도우를 포함하는 그래픽 사용자 인터페이스 구성 요소 조작 방법.

청구항 32

컴퓨터 디스플레이 상의 다수개의 윈도우를 조작하는 방법에 있어서,

a) 다수개의 윈도우를 컴퓨터 디스플레이 상에 디스플레이하는 단계;

b) 포인터를 컴퓨터 디스플레이 상에 디스플레이하는 단계;

c) 다수개의 윈도우로부터 제1 및 제2 윈도우들을 실행 윈도우 - 상기 실행 윈도우는 포인터에 근접 위치되는 원점 둘레에 정의되는 근접 범위 내에 적어도 부분적으로 배치하는 것임-로서 정의하는 단계; 및

d) 사용자 입력에 응답하여 원점에 대해 내부 및 외부 방향 중 한 방향으로 각 실행 윈도우의 적어도 일부를 동시에 이동시키는 단계

를 포함하는 윈도우 조작 방법.

청구항 33

제32항에 있어서, 각 윈도우는 적어도 하나의 경계 세그먼트를 구비한 경계를 포함하고, 각 실행 윈도우는 근접 범위 내에 적어도 부분적으로 배치되는 실행 경계 세그먼트를 포함하며, 상기 각 실행 윈도우의 적어도 일부를 동시에 이동시키는 단계는 상기 실행 윈도우들의 실행 경계 세그먼트들을 적어도 동시에 이동시키는 단계를 포함하는 윈도우 조작 방법.

청구항 34

제33항에 있어서, 상기 실행 윈도우들의 실행 경계 세그먼트들을 적어도 동시에 이동시키는 단계는 상기 실행 경계 세그먼트를 원점에 대해 방사상으로 이동시키는 단계를 포함하는 윈도우 조작 방법.

청구항 35

제33항에 있어서, 상기 실행 윈도우들의 실행 경계 세그먼트들을 적어도 동시에 이동시키는 단계는,

a) 제1 사용자 입력에 응답하여 원점을 향해 내부 방향으로 상기 실행 경계 세그먼트를 이동시키는 단계; 및

b) 제2 사용자 입력에 응답하여 원점으로부터 외부 방향으로 상기 실행 경계 세그먼트들을 이동시키는 단계;

를 포함하는 윈도우 조작 방법.

청구항 36

제33항에 있어서, 각 실행 윈도우의 적어도 일부를 이동시키는 단계가 실행 윈도우를 이동시키는 단계를 포함하는 윈도우 조작 방법.

청구항 37

컴퓨터 시스템에 있어서,

a) 복수개의 윈도우 및 포인터가 디스플레이되는 컴퓨터 디스플레이;

b) 사용자 입력을 수신하도록 구성되는 사용자 입력 장치; 및

c) 상기 컴퓨터 디스플레이 및 사용자 입력 장치에 결합되고, 다수개의 윈도우로부터 제1 및 제2 윈도우들을 실행 윈도우 - 상기 실행 윈도우는 적어도 부분적으로는 포인터에 인접한 원점 둘레에 정의되는 근접 범위 내에 배치되는 것임 - 로서 정의하며, 사용자 입력에 응답하여 원점에 대해 내부 및 외부 방향 중 어느 한 방향으로 각 실행 윈도우의 적어도 일부를 동시에 이동시키도록 구성되는 프로세서를

포함하는 컴퓨터 시스템.

청구항 38

i) 다수개의 윈도우를 컴퓨터 디스플레이상에 디스플레이하는 단계;

ii) 포인터를 컴퓨터 디스플레이 상에 디스플레이하는 단계;

iii) 다수개의 윈도우로부터의 제1 및 제2 윈도우들을 실행 윈도우 - 상기 실행 윈도우는 적어도 부분적으로는 포인터에 인접한 원점 둘레에 정의되는 근접 범위 내에 배치되는 것임 - 로서 정의하는 단계; 및

iv) 사용자 입력에 응답하여 원점에 대해 내부 및 외부 방향 중 어느 한 방향으로 각 실행 윈도우의 적어도 일부를 동시에 이동시키는 단계;

를 포함하는 컴퓨터 디스플레이 상에서 다수개의 윈도우를 조작하는 방법을 수행하도록 구성되는 컴퓨터 프로그램을 포함한 컴퓨터 판독가능한 기록 매체.

청구항 39

컴퓨터 디스플레이 상에서 다수개의 그래픽 사용자 인터페이스 구성 요소를 조작하는 방법에 있어서,

a) 다수개의 그래픽 사용자 인터페이스 구성 요소를 컴퓨터 디스플레이 상에 디스플레이하는 단계;

b) 포인터를 컴퓨터 디스플레이 상에 디스플레이하는 단계;

c) 다수의 그래픽 사용자 인터페이스 구성 요소로부터 제1 및 제2 그래픽 사용자 인터페이스 구

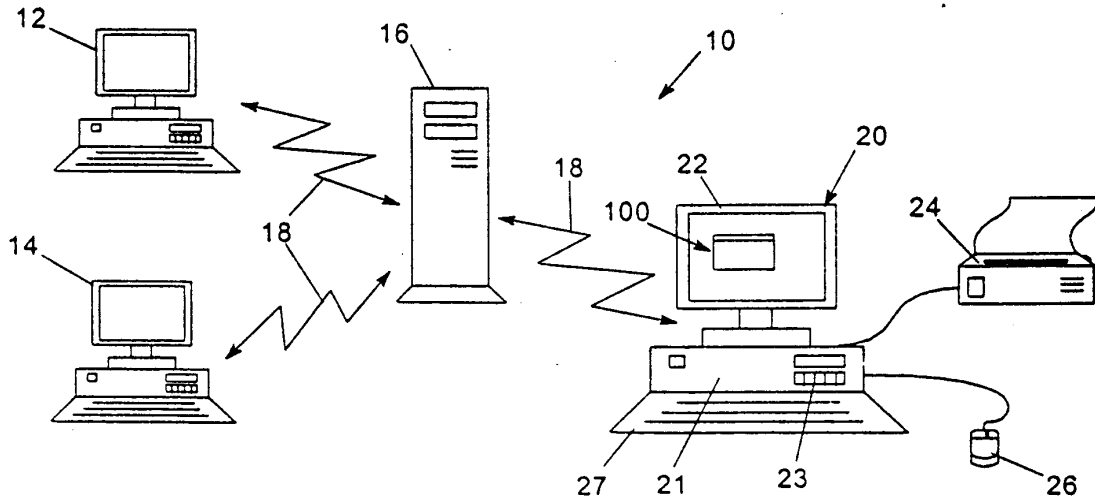
성 요소들을 실행 그래픽 사용자 인터페이스 구성 요소 - 상기 실행 그래픽 사용자 인터페이스 구성 요소는 포인터에 인접한 원점 둘레에 정의되는 근접 범위 내에 배치되는 것임 - 로서 정의하는 단계; 및

d) 사용자 입력에 응답하여, 원점에 대해 내부 및 외부 방향 중 한 방향으로 각 실행 그래픽 사용자 인터페이스 구성 요소의 적어도 일부를 동시에 이동시키는 단계

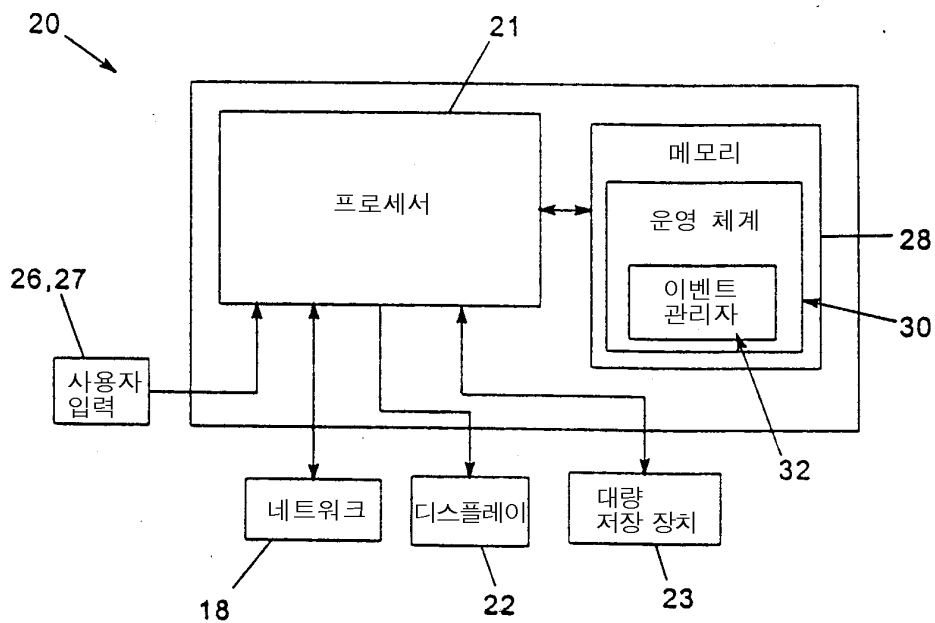
를 포함하는 그래픽 사용자 인터페이스 구성 요소 조작 방법.

도면

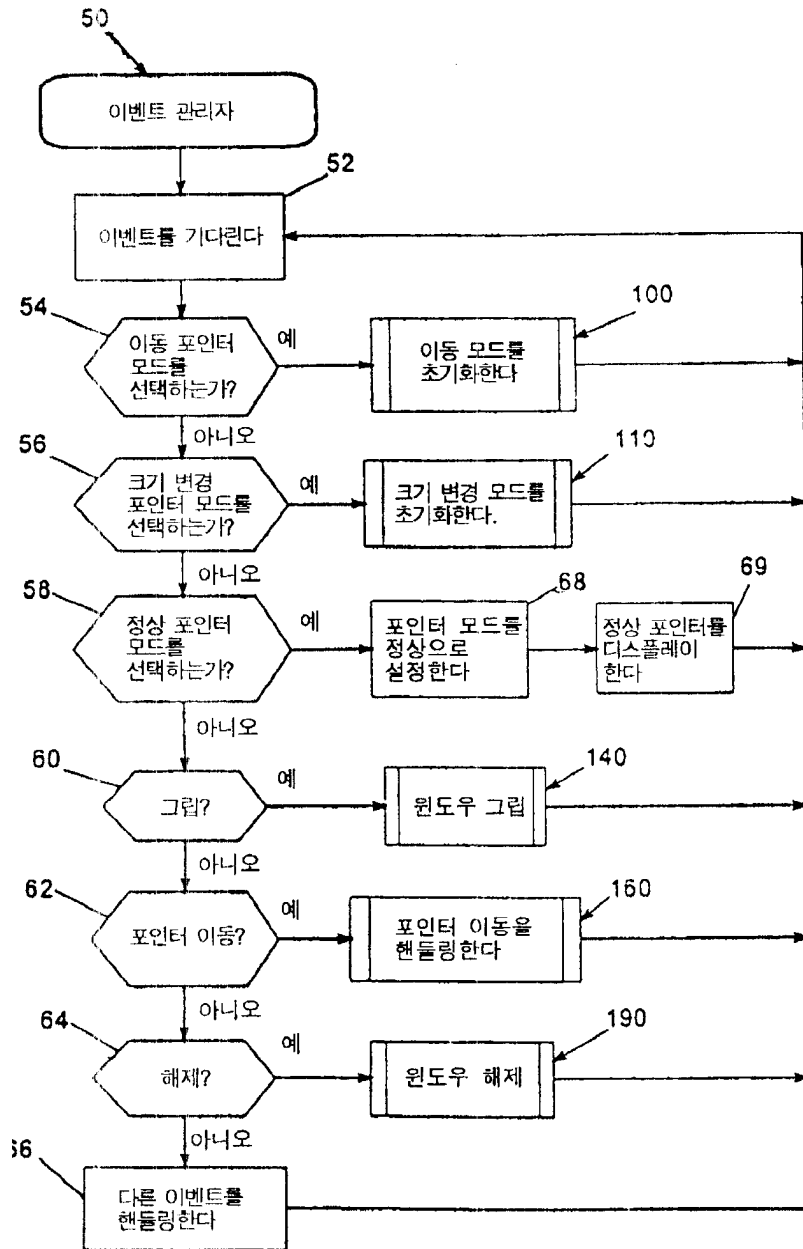
도면1



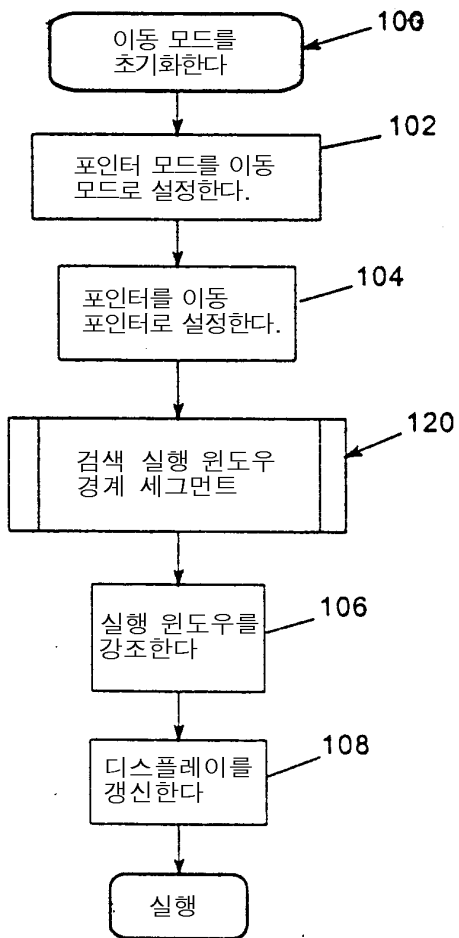
도면2



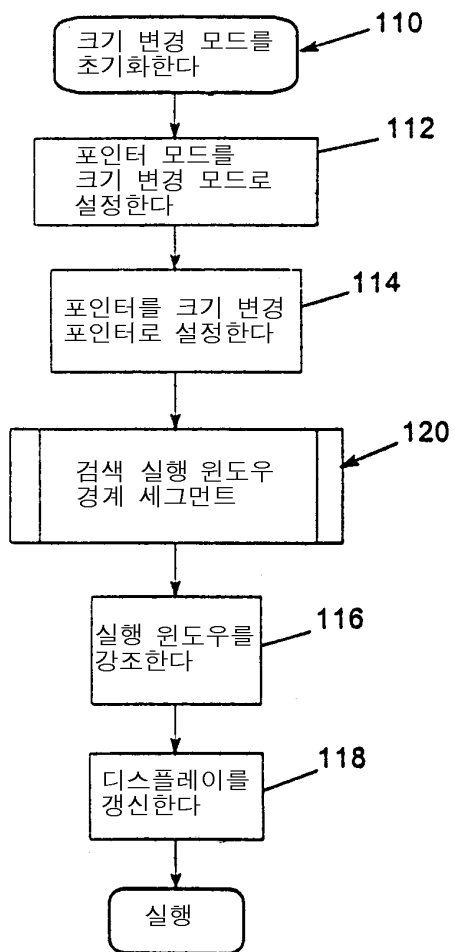
도면3



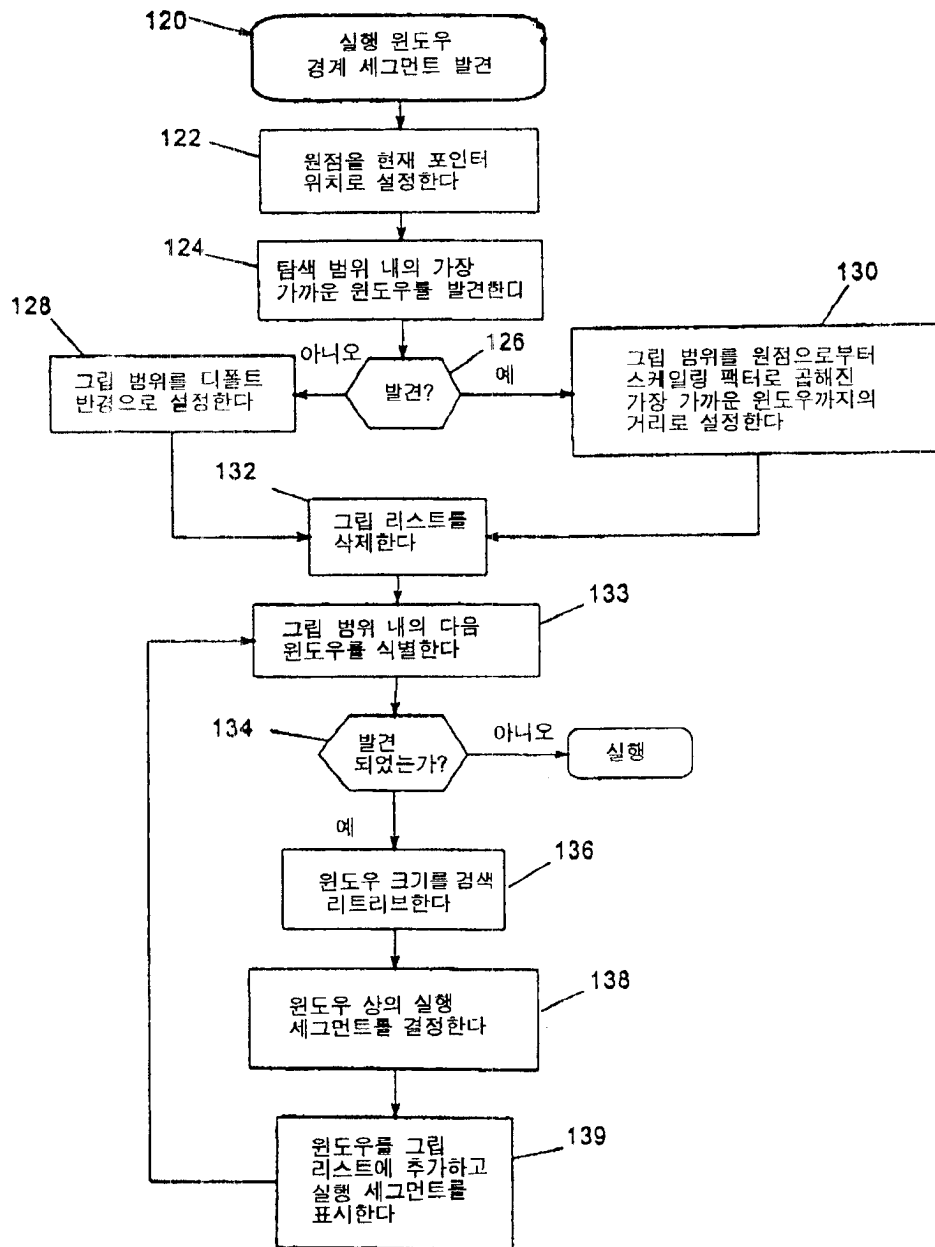
도면4



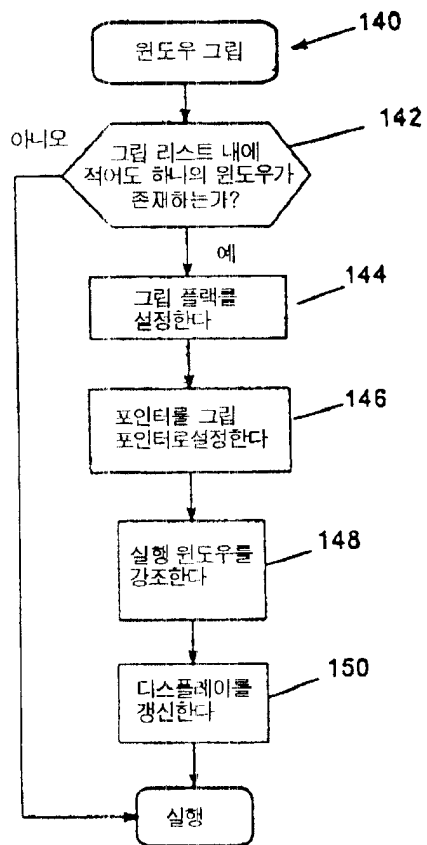
도면5



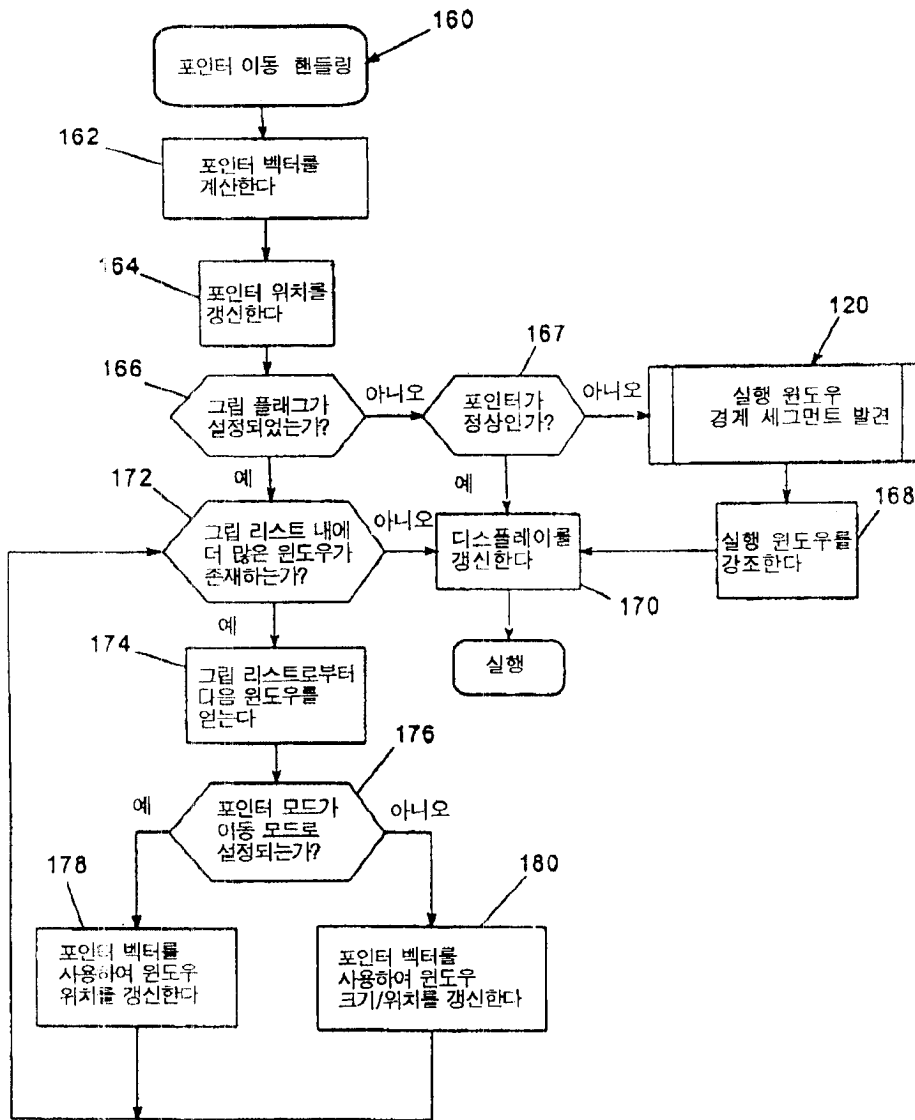
도면6



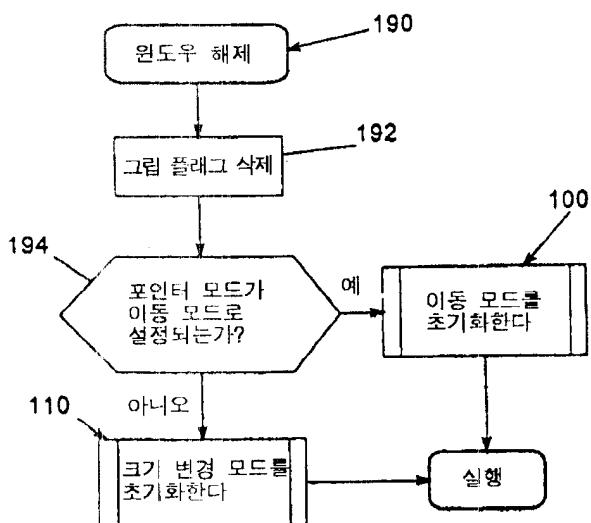
도면7



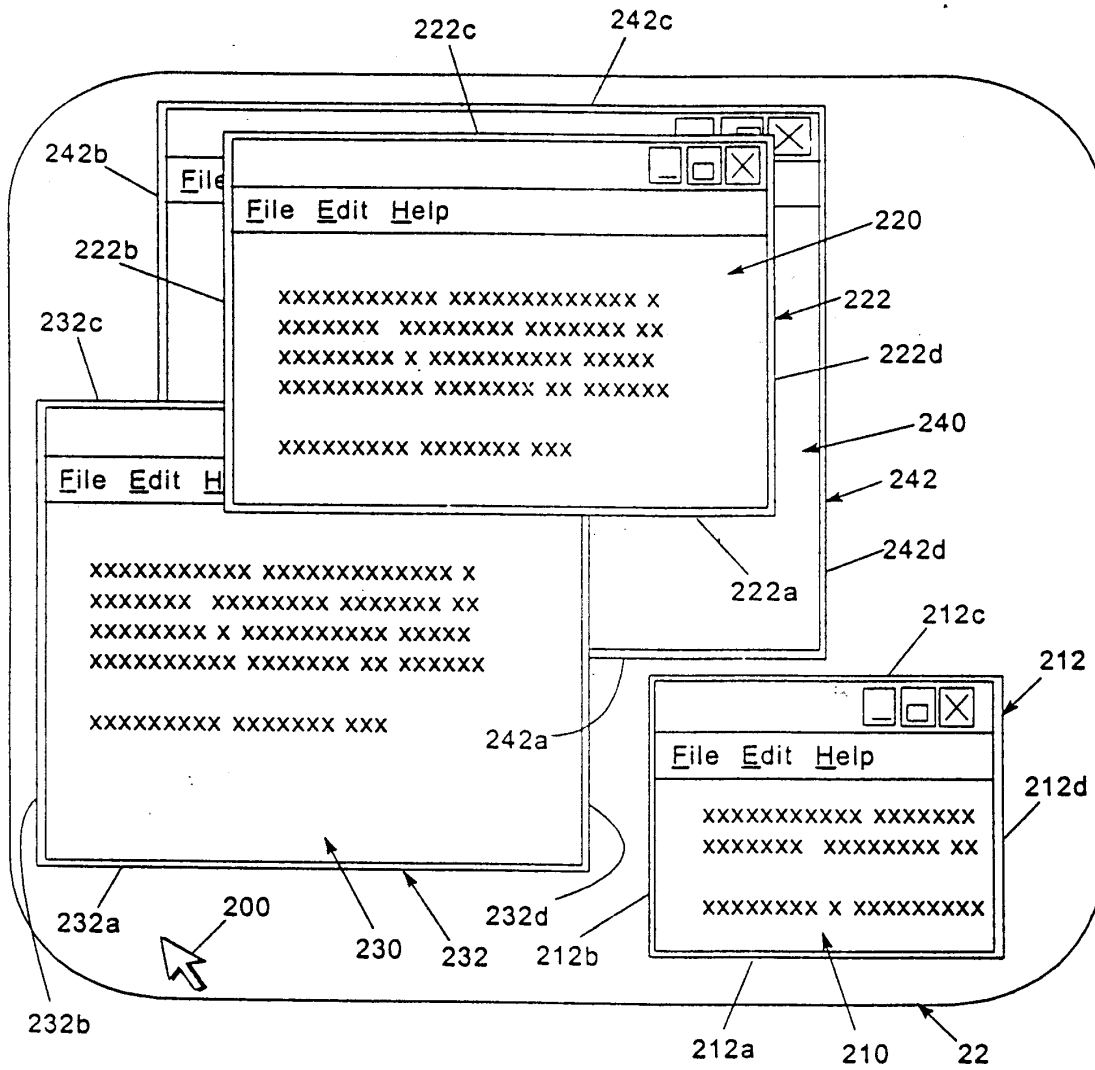
도면8



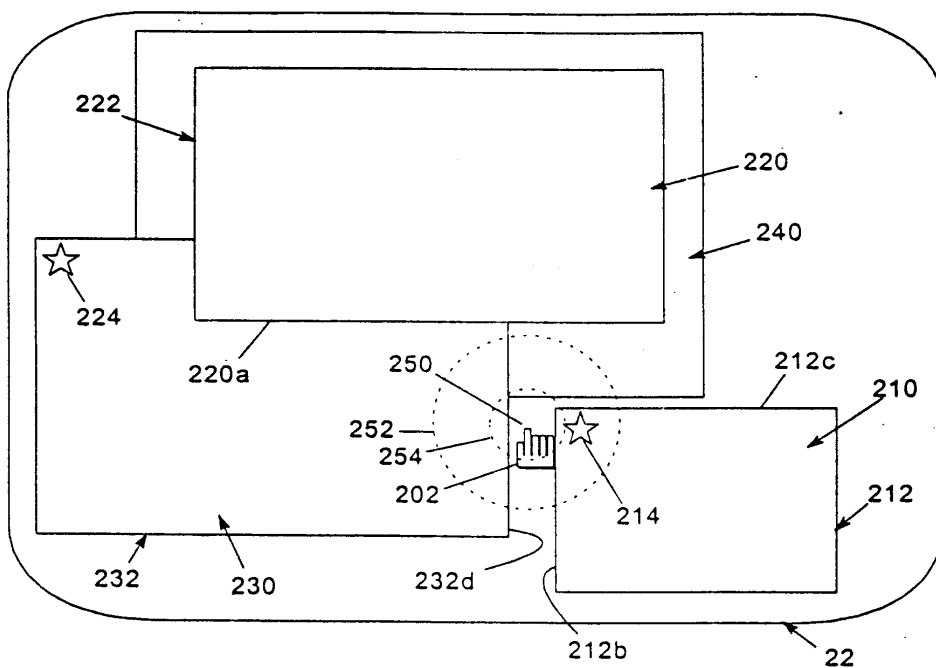
도면9



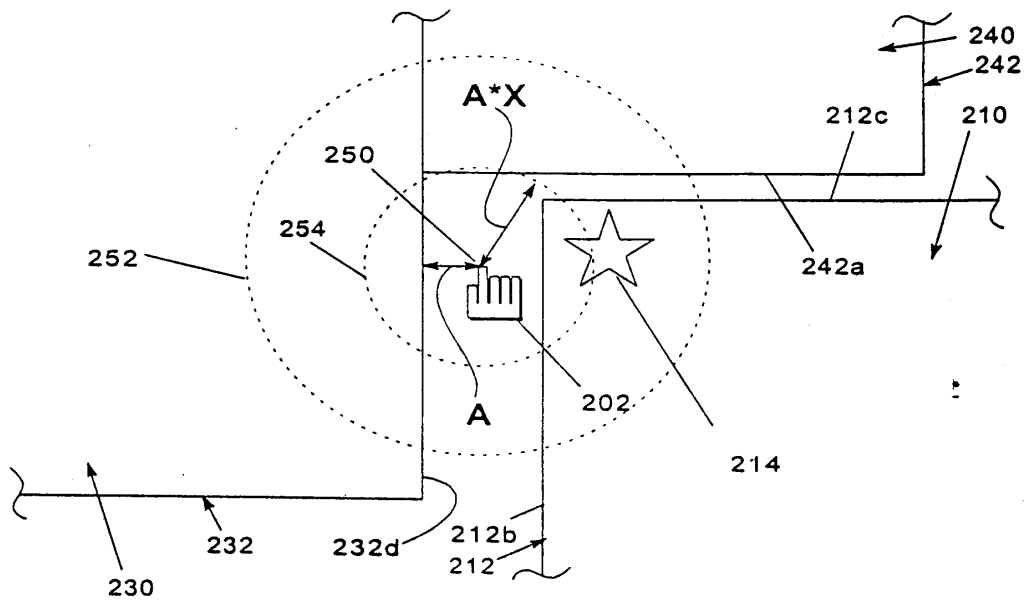
도면10



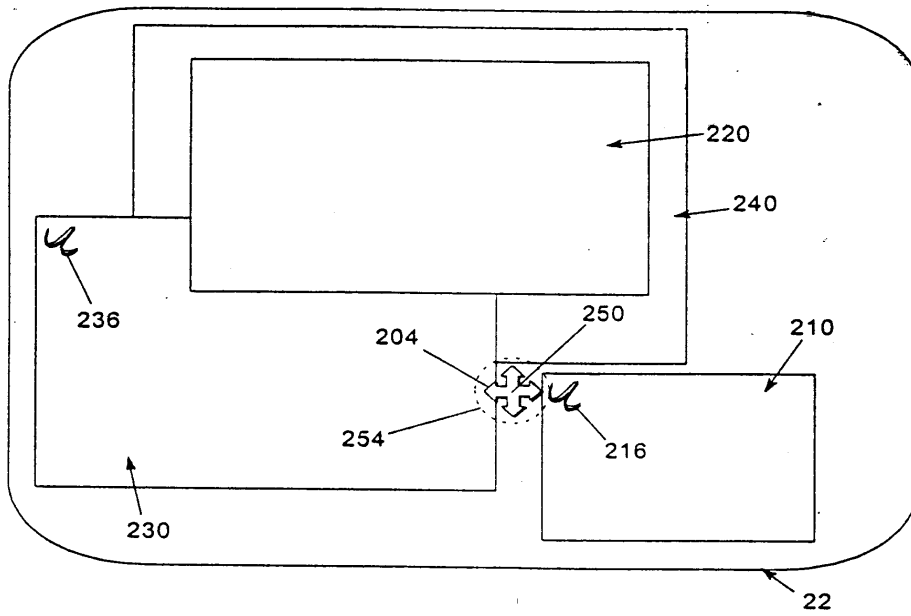
도면11



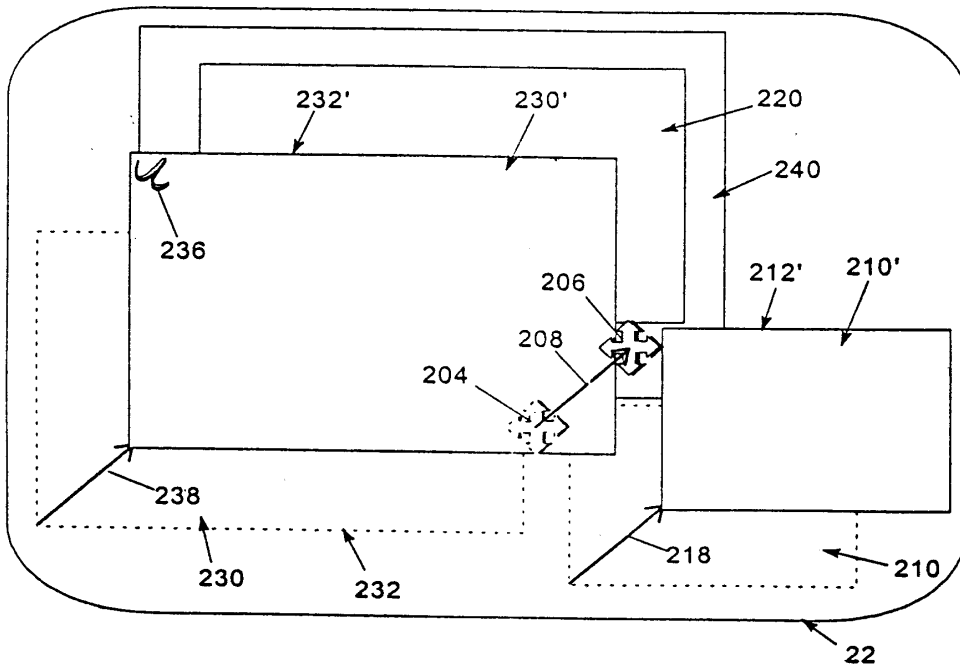
도면 12



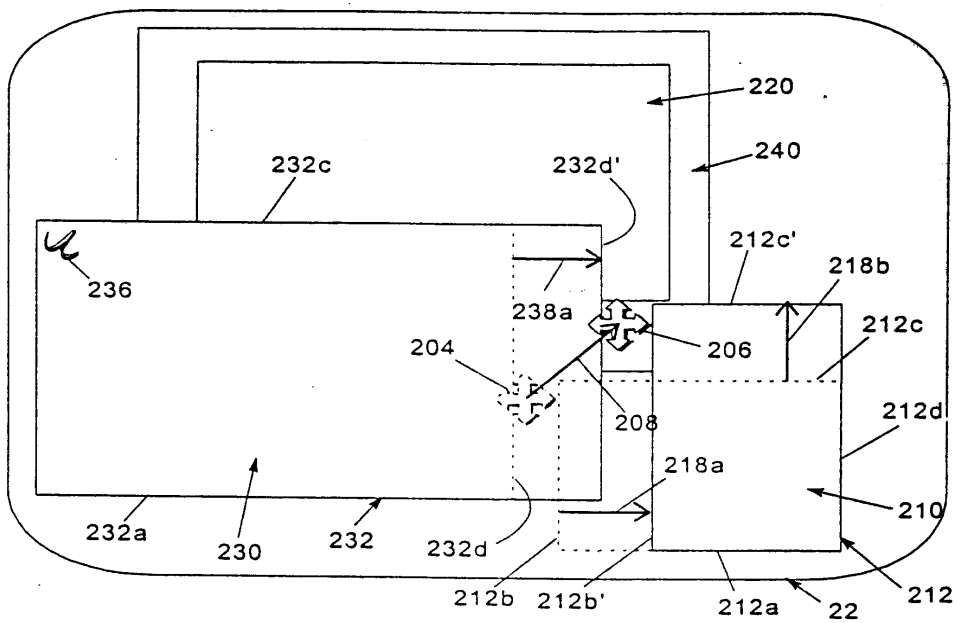
도면 13



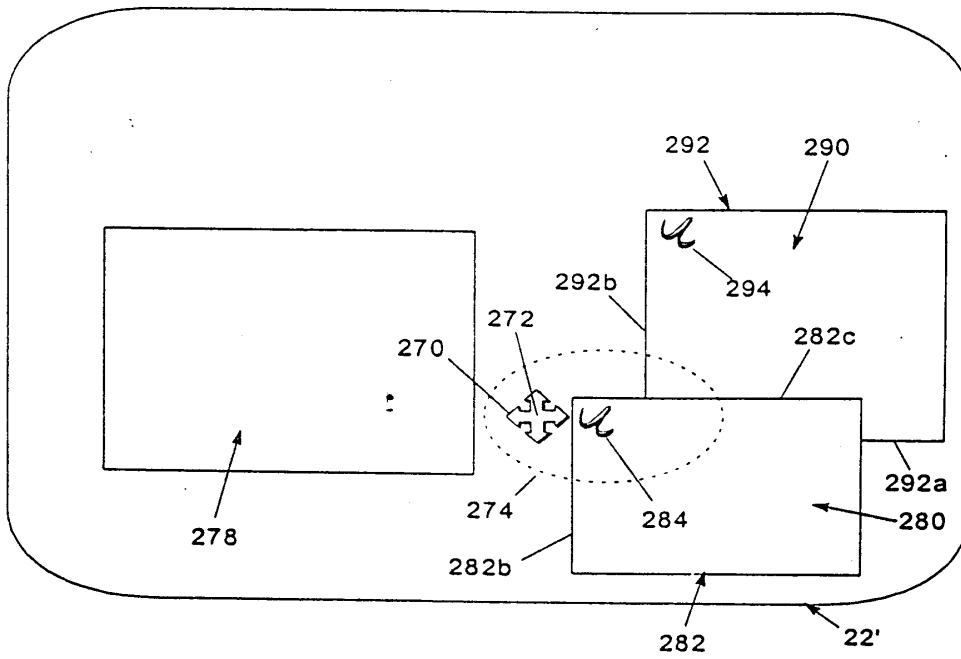
도면14



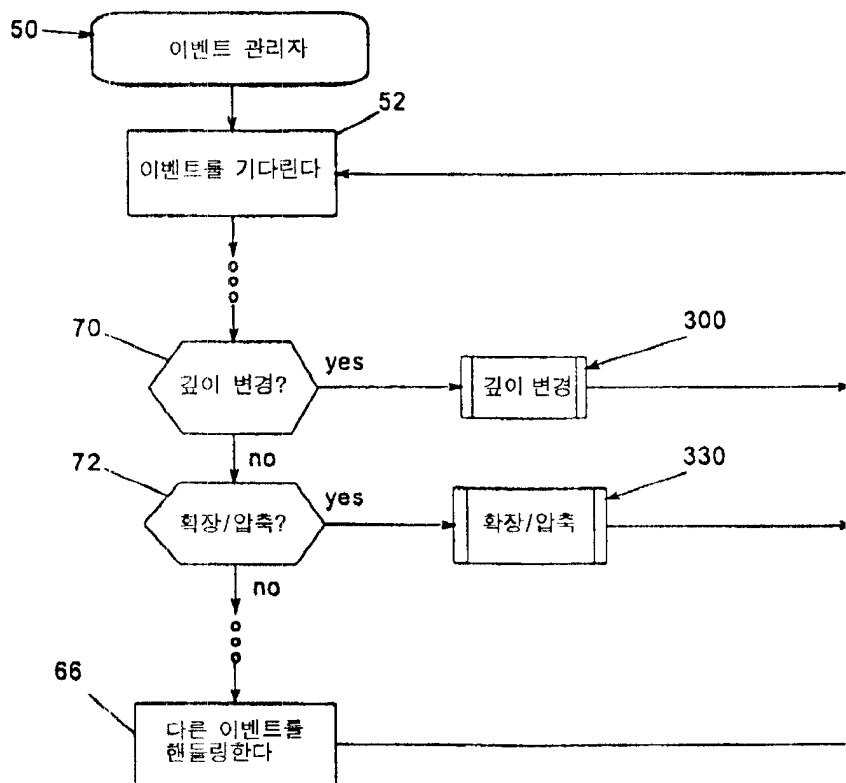
도면15



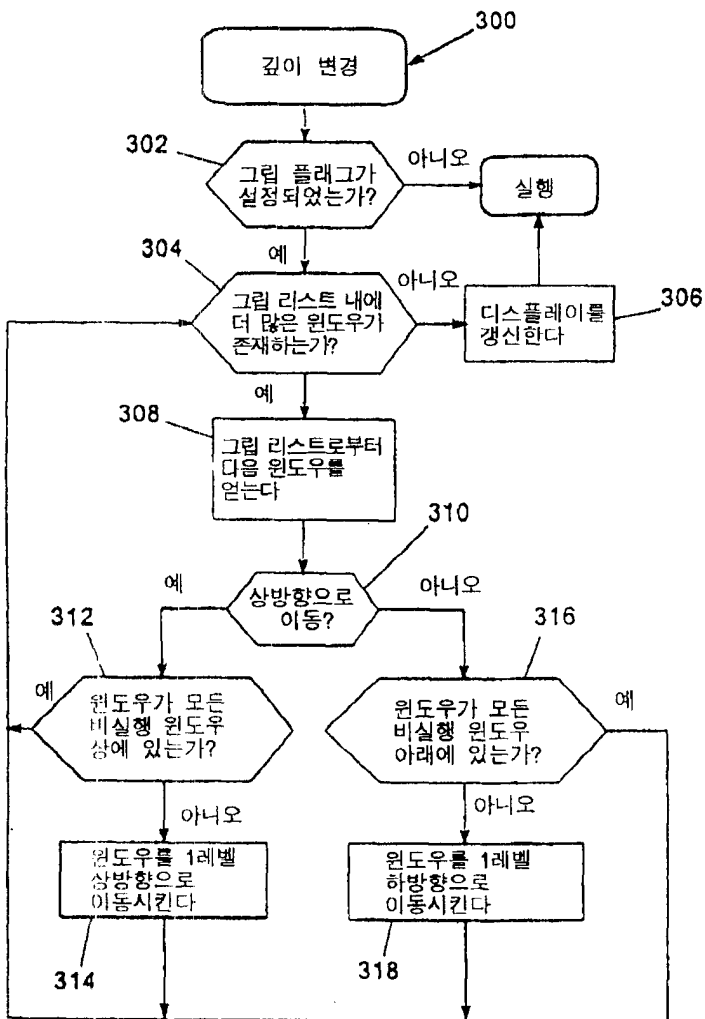
도면16



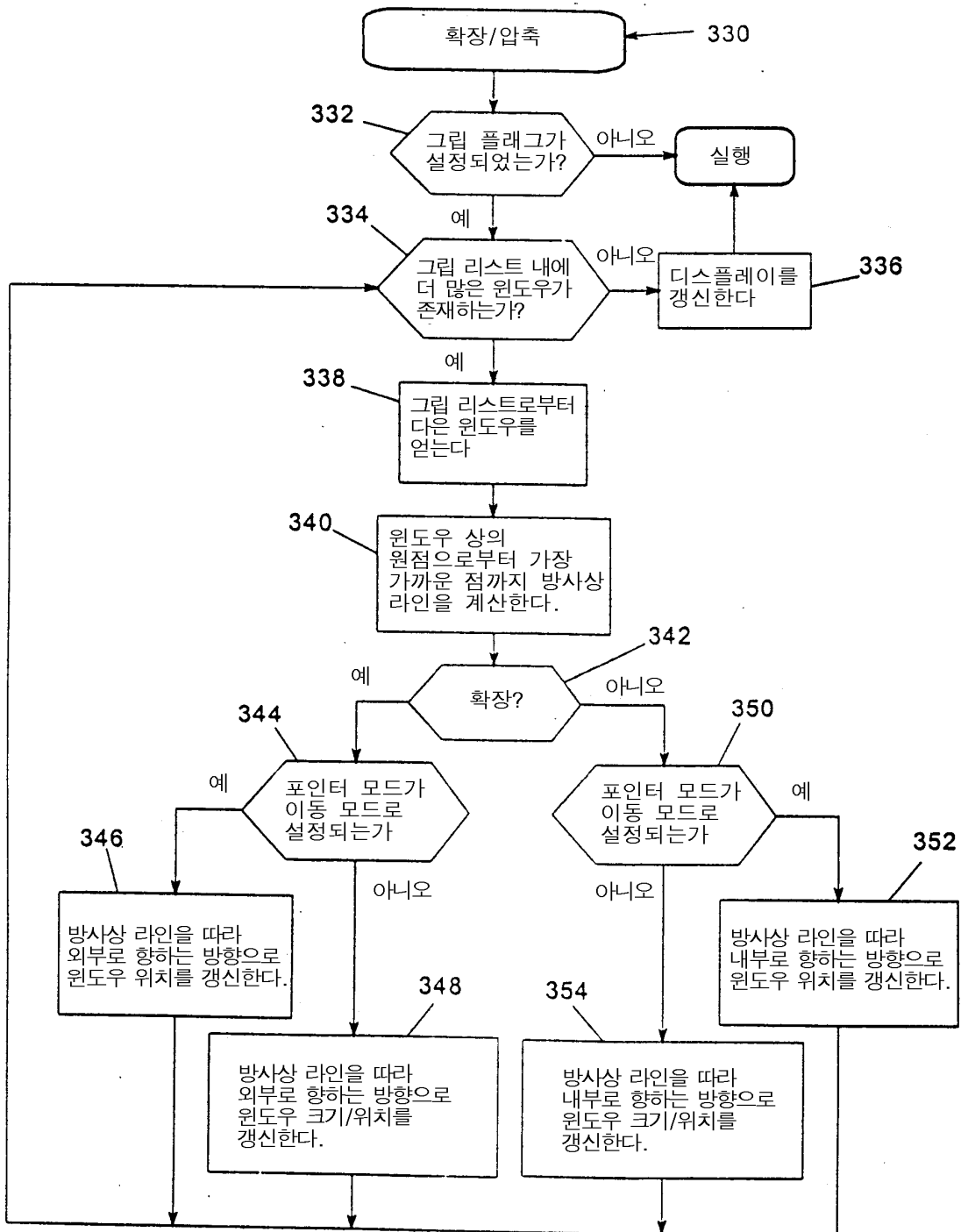
도면17



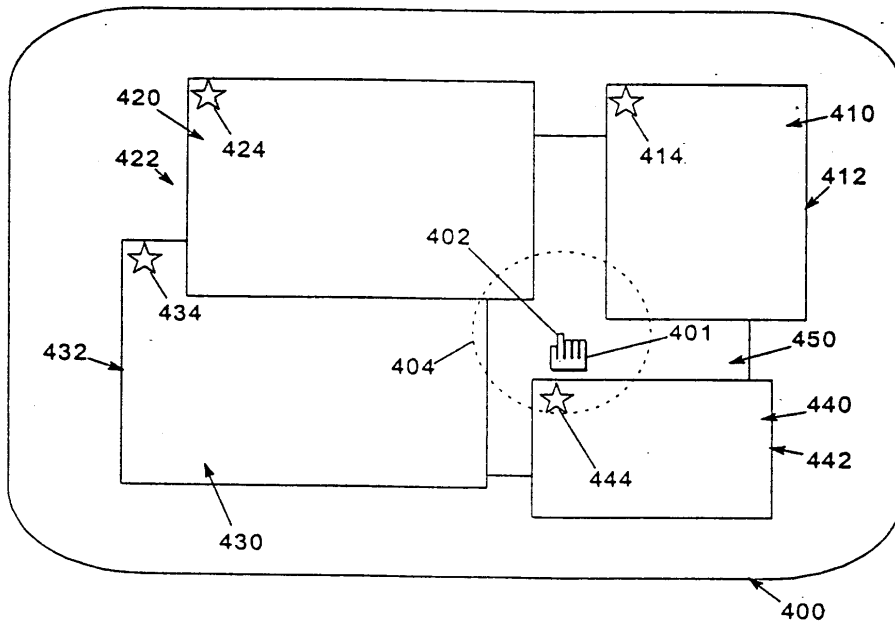
도면 18



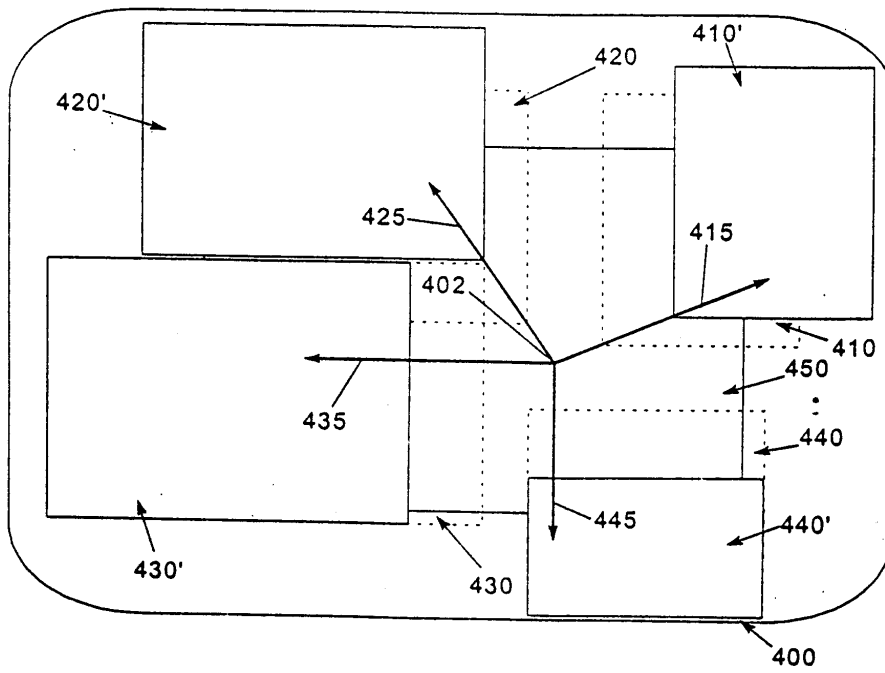
도면 19



도면20



도면21



도면22

