(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
17 February 2011 (17.02.2011)

PCT

(10) International Publication Number
WO 2011/019615 A1

(51) International Patent Classification:
G06F 1/20 (2006.01)

(21) International Application Number:
PCT/US2010/044740

(22) International Filing Date:
6 August 2010 (06.08.2010)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
12/540,034 12 August 2009 (12.08.2009) US

(71) Applicant *(for all designated States except US)*: **INTERNATIONAL BUSINESS MACHINES CORPORATION** [US/US]; New Orchard Road, Armonk, New York 10504 (US).

(72) Inventors; and

(75) Inventors/Applicants *(for US only)*: **HAMANN, Hendrik F.** [DE/US]; IBM T.J. Watson Research Center, Route 134, Yorktown Heights, New York 10598 (US). **LOPEZ-MARRERO, Vanessa** [US/US]; IBM T.J. Watson Research Center, Route 134, Yorktown Heights, New York 10598 (US). **STEPANCHUCK, Andrew** [UA/US]; 404 Sterling Dr., Wappingers Falls, New York 12590 (US).

(74) Agent: **CHANG, Michael J.**; 84 Summit Avenue, Milford, Connecticut 06460 (US).

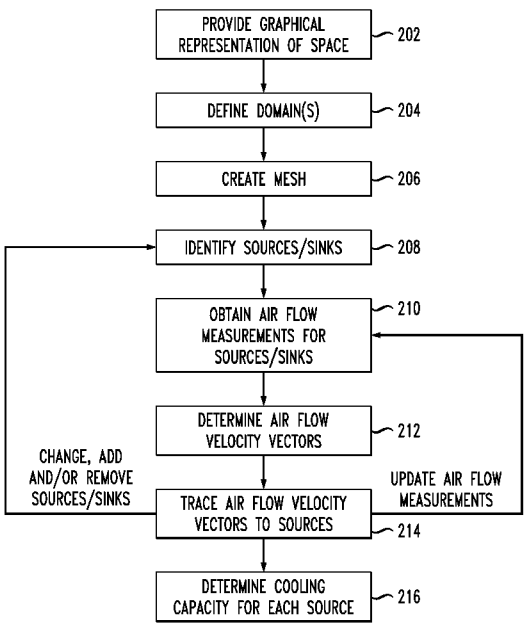
(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IS, JP, KE, KG, KM, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LT, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PE, PG, PH, PL, PT, RO, RS, RU, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU,

[Continued on next page]

(54) Title: METHODS AND TECHNIQUES FOR CREATING AND VISUALIZING THERMAL ZONES

FIG. 2



(57) Abstract: Techniques for using air flow analysis to model thermal zones are provided. In one aspect, a method for modeling thermal zones in a space, e.g., in a data center, includes the following steps. A graphical representation of the space is provided. At least one domain is defined in the space for modeling. A mesh is created in the domain by sub-dividing the domain into a set of discrete sub-domains that interconnect a plurality of nodes. Air flow sources and sinks are identified in the domain. Air flow measurements are obtained from one or more of the air flow sources and sinks. An air flow velocity vector at a center of each sub- domain is determined using the air flow measurements obtained from the air flow sources and sinks. Each velocity vector is traced to one of the air flow sources, wherein a combination of the traces to a given one of the air flow sources represents a thermal zone in the space.

WO 2011/019615 A1

LV, MC, MK, MT, NL, NO, PL, PT, RO, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Declarations under Rule 4.17:

— *as to applicant's entitlement to apply for and be granted a patent (Rule 4.17(ii))*

— *as to the applicant's entitlement to claim the priority of the earlier application (Rule 4.17(iii))*

Published:

— *with international search report (Art. 21(3))*

METHODS AND TECHNIQUES FOR CREATING AND VISUALIZING THERMAL ZONES

Field of the Invention

[0001] The present invention relates to air flow analysis, and more particularly, to techniques for using air flow distributions to model thermal zones in a space, such as in a data center.

Background of the Invention

[0002] Energy consumption has become a critical issue for large scale computing facilities (or data centers), triggered by the rise in energy costs, supply and demand of energy and the proliferation of power hungry information and communication technology (ICT) equipment. Data centers consume approximately two percent (%) of all electricity globally or 183 billion kilowatt (KW) hours of power; this power consumption is growing at a rate of 12% each year. A significant fraction of the power consumption, i.e., up to 50%, is directed to cooling the heat generating equipment. Consequently, the improvement of data center energy and cooling efficiency is very important. Although best practices have been widely publicized, data center operators are struggling to provision the right amount of cooling. In particular, it is challenging to take different heat densities within a data center into account (i.e., different areas within the data center may require very different amounts of cooling).

[0003] Therefore techniques directed to highlighting heat densities within a data center and thereby increasing cooling efficiency would be desirable.

Summary of the Invention

[0004] The present invention provides techniques for using air flow distributions to model thermal zones. In one aspect of the invention, a method for modeling thermal zones in a space, e.g., in a data center, is provided. The method includes the following steps. A graphical representation of the space is provided. At least one domain is defined in the space for modeling. A mesh is created in the domain by sub-dividing the domain into a set of discrete sub-domains that interconnect a plurality of nodes. Air flow sources and sinks are identified in the domain. Air flow measurements are obtained from one or more of the air flow sources and sinks. An air flow velocity vector at a center of each sub-domain is determined using the air flow measurements obtained from the air flow sources and sinks.

Each velocity vector is traced to one of the air flow sources, wherein a combination of the traces to a given one of the air flow sources represents a thermal zone in the space.

[0005] A more complete understanding of the present invention, as well as further features and advantages of the present invention, will be obtained by reference to the following detailed description and drawings.

Brief Description of the Drawings

[0006] FIG. 1 is a diagram illustrating an exemplary data center according to an embodiment of the present invention;

[0007] FIG. 2 is a diagram illustrating an exemplary methodology for modeling thermal zones according to an embodiment of the present invention;

[0008] FIG. 3 is a diagram illustrating an exemplary finite element mesh according to an embodiment of the present invention;

[0009] FIG. 4 is a diagram illustrating a graphical representation of a data center according to an embodiment of the present invention;

[0010] FIG. 5 is a diagram illustrating an enlarged view of a portion of the graphical representation of FIG. 4 showing a mesh according to an embodiment of the present invention;

[0011] FIG. 6 is a diagram illustrating an enlarged view of the portion of the graphical representation of FIG. 4 showing air flow potential and air flow velocity vectors according to an embodiment of the present invention;

[0012] FIG. 7 is a diagram illustrating an enlarged view of the portion of the graphical representation of FIG. 4 showing traces and corresponding thermal zones according to an embodiment of the present invention;

[0013] FIG. 8 is a diagram illustrating an exemplary graphical interface containing a representation of a data center according to an embodiment of the present invention;

[0014] FIG. 9 is a diagram illustrating how the graphical interface of FIG. 8 can be edited to reflect changes in the data center according to an embodiment of the present invention;

[0015] FIG. 10 is a diagram illustrating the visualization of thermal zones on the graphical interface of FIG. 8 according to an embodiment of the present invention;

[0016] FIG. 11 is a schematic diagram illustrating cooling capacity in a data center according to an embodiment of the present invention;

[0017] FIG. 12 is a diagram illustrating an exemplary methodology for tracing air flow velocity vectors to an air flow source according to an embodiment of the present invention; and

[0018] FIG. 13 is a diagram illustrating an exemplary apparatus for modeling thermal zones in a space according to an embodiment of the present invention.

Detailed Description of Preferred Embodiments

[0019] Presented herein are techniques for dynamically creating and visualizing thermal zones, which enables better provisioning and a more efficient usage of cooling for data centers. It is notable that while the instant techniques are described in the context of a cooling system of a data center, the concepts presented herein are generally applicable to cooling and/or heating systems in general.

[0020] FIG. 1 is a diagram illustrating exemplary data center 100. Data center 100 has server racks 101 and a raised-floor cooling system with air conditioning units (ACUs) 102 (which may also be referred to as computer room air conditioners (CRACs)) that take hot air in (typically from above through one or more air returns in the ACUs) and exhaust cooled air into a sub-floor plenum below. Hot air flow through data center 100 is indicated by light arrows 110 and cooled air flow through data center 100 is indicated by dark arrows 112. In the following description, the data center above the sub-floor plenum may also be referred to simply as the raised floor, and the sub-floor plenum may be referred to simply as the plenum. Thus, by way of example only, as shown in FIG. 1, the ACUs intake warm air from the raised floor and expel cooled air into the plenum (see below).

[0021] In FIG. 1, server racks 101 use front-to-back cooling and are located on raised-floor 106 with sub-floor 104 beneath. Namely, according to this scheme, cooled air is drawn in through a front (inlet) of each rack and warm air is exhausted out from a rear (outlet) of each rack. The cooled air drawn into the front of the rack is supplied to air inlets of each information technology (IT) equipment component (servers for example) therein. Space between raised floor 106 and sub-floor 104 defines the sub-floor plenum 108. The sub-floor plenum 108 serves as a conduit to transport, e.g., cooled air from the ACUs 102 to the racks. As shown in FIG. 1, raised-floor 106 consists of a plurality of floor tiles, some of which are

perforated. In a properly-organized data center (such as data center 100), racks 101 are arranged in a hot aisle – cold aisle configuration, i.e., having air inlets and exhaust outlets in alternating directions. Namely, cooled air is blown through the perforated floor tiles 114 (also referred to as vents) in raised-floor 106, from the sub-floor plenum 108 into the cold aisles. The cooled air is then drawn into racks 101, via the air inlets, on an air inlet side of the racks and dumped, via the exhaust outlets, on an exhaust outlet side of the racks and into the hot aisles.

[0022] The ACUs typically receive chilled water from a refrigeration chiller plant (not shown). Each ACU typically comprises a blower motor to circulate air through the ACU and to blow cooled air, e.g., into the sub-floor plenum. As such, in most data centers, the ACUs are simple heat exchangers mainly consuming power needed to blow the cooled air into the sub-floor plenum. Typically, one or more power distribution units (PDUs) (not shown) are present that distribute power to the server racks 101.

[0023] It is important to optimize the efficiency of the ACUs. See, for example, Hamann et al., "Uncovering Energy-Efficiency Opportunities in Data Centers," IBM Journal of Research and Development, vol. 53, no. 3 (2009) (hereinafter "Hamann"), the contents of which are incorporated by reference herein. To this end, it is useful to consider the utilization levels of the ACUs (i.e., utilization (UT) = heat removed / nominal heat load removal capacity) or coefficient of performance (COP) (COP=heat removed / power consumption for ACU fans). It has been shown that utilization levels of ACUs in some data centers are as low as 10% (COP ~ 1.8). However, utilization levels could potentially be in the range of from about 80% to about 100% (even with some redundancy) with corresponding COPs of from about 14 to about 18, respectively, if efficiency optimization practices are employed. Most data centers require some redundancy. For example, in a data center with eight ACUs, redundancy allows for one ACU to fail (for example due to mechanical failure). So, in essence the target utilization level cannot be 7/8 (87.5%) because then there would not be N+1 redundancy (N=8).

[0024] One of the inhibitors to optimizing ACU usage is a lack of visibility, i.e., an inability to discern to which physical areas or zones in a data center different ACUs are supplying cooled air. Thermal zones are physical areas (two-dimensional (2D)) or volumes (three-dimensional (3D)) within the data center. Each ACU supplies air to a specific thermal zone within the data center, which may also be referred to herein as a "supply zone" and typically applies to the plenum. Each ACU also gets return air from a specific thermal zone in the data

center, which may also be referred to herein as a "return zone" and typically applies to the raised floor.

[0025] It is not unusual for a data center to have more than 50 ACUs somewhat randomly distributed across the data center space. The thermal zone resulting from each ACU (supply and/or return zone) is not only governed by the air flow produced by each ACU but also by the placement of vents or perforated tiles throughout the data center (because the vents or perforated tiles direct, at least in part, to where in the data center the air from the ACUs is directed, see below). It is not unusual for data centers to have more than 1,000 vents or perforated tiles. Because these thermal zones are based on the actual air flow contribution of each ACU, a corresponding efficiency or coefficient of performance (COP) can be assigned to each thermal zone of each respective ACU. The air flow distribution throughout a data center is governed by many aspects. The present techniques use the simplest form of air flow distribution in a data center by applying the zone concept to the plenum. ACUs discharge air into the plenum using fans and as a result the plenum is pressurized. The placement of the vents or perforated tiles governs where the air escapes the plenum on the raised floor. That vent/perforated tile placement determines the zones (i.e., which area is supplied by which ACU).

[0026] Disclosed herein are techniques for modeling, i.e., creating and visualizing, these thermal zones. As will be described in further detail below, a velocity field is used to define, i.e., create, the thermal zones (that is, the zones are not defined beforehand, and thus must be created). Creating/defining the zones is an important aspect of the present techniques as it provides one way to determine how efficiently each ACU is being used (via the COP measure). FIG. 2 is a diagram illustrating exemplary methodology 200 for modeling thermal zones in a space, e.g., within a room, such as data center 100. In step 202, a graphical representation of the space is provided. For example, a two-dimensional graphical representation of a data center is provided in FIG. 4, described below. In the case of a data center, the graphical representation can include, for example, the layout of the server racks, the vents or perforated tiles and the ACUs. The graphical representation can be created using a software application, which "draws" the assets such as the server racks. An example for this is a mobile management technology software application (which is what was used to create the graphical representation shown in FIG. 4).

[0027] In step 204, at least one domain for thermal zone modeling is defined in the space. Each domain can be a two-dimensional or three-dimensional domain. As will be described

below, in one exemplary embodiment wherein a data center is being modeled, the domain is defined by the dimensions of the sub-floor plenum. As will also be described below, more than one domain can be defined for a given space. Depending on the particular application, such as the physical layout of the space to which the present techniques are applied, the domain(s) may comprise the entire space, or a portion(s) thereof.

[0028] In step 206, since finite elements are being employed to model the space, a finite element mesh is created in each of the domains by sub-dividing each domain into a set of discrete sub-domains that interconnect a plurality of nodes. As will be described in detail below, the sub-domains (also referred to herein as "elements") can be triangles (in the case of two-dimensional domains) or tetrahedra (in the case of three-dimensional domains). The use of triangles and tetrahedra are standard choices in the finite element method. The nodes correspond to x and y coordinates (in the case of two-dimensional domains) or to x, y and z coordinates (in the case of three-dimensional domains).

[0029] In step 208, air flow sources (where the airflow enters the domain) and air flow sinks (where the air flow exits the domain) are identified in the domain. By way of reference to a data center, when the domain includes the sub-floor plenum the perforated tiles can be considered air flow sinks (because it is at the perforated tiles where the cooled air supplied by the ACUs exits the sub-floor plenum and enters the raised floor) and the ACUs can be considered air flow sources (because the airflow originates/enters the sub-floor plenum from the ACUs). On the other hand, when the domain includes the raised floor the perforated tiles can be considered air flow sources (because it is at the perforated tiles where the cooled air from the plenum enters the raised floor) and the ACUs can be considered sinks (because the warm air is cycled back into the ACUs and thus exits the raised-floor at the ACUs).

[0030] In step 210, air flow measurements are obtained from one or more of the air flow sources and sinks. According to an exemplary embodiment, these air flow measurements are obtained using mobile measurement technology (MMT). MMT is described for example in U.S. Patent Number 7,366,632, issued to Hamann et al., entitled "Method and Apparatus for Three-Dimensional Measurements" (hereinafter "U.S. Patent Number 7,366,632") the contents of which are incorporated by reference herein. MMT V1.0 is a technology for optimizing data center infrastructures for improved energy and space efficiency which involves a combination of advanced metrology techniques for rapid measuring/surveying data centers (see, for example, U.S. Patent Number 7,366,632) and metrics-based assessments and data-based best practices implementation for optimizing a data center within a given thermal

envelope for optimum space and most-efficient energy utilization (see, for example, U.S. Application Serial Number 11/750,325, filed by Claassen et al., entitled "Techniques for Analyzing Data Center Energy Utilization Practices," designated as Attorney Reference Number YOR920070242US1, the contents of which are incorporated by reference herein).

[0031] In step 212, the air flow measurements obtained from the air flow sources and sinks are used to determine an air flow velocity vector at a center of each sub-domain (element). An exemplary process for determining these air flow velocity vectors using potential flow theory is described in detail below.

[0032] In step 214, each velocity vector is traced to one of the air flow sources. By way of reference to a data center model wherein the domain comprises the sub-floor plenum, each velocity vector can be traced to a particular ACU. An exemplary process for tracing the velocity vectors is described in detail below. A combination of the traces to a given one of the air flow sources (e.g., ACU) represents a thermal zone in the space. Exemplary thermal zones are shown, for example, in FIG. 10, described below.

[0033] As shown in FIG. 2, steps 208-214 of methodology 200 can be repeated in response to changes in the air flow at one or more of the sources/sinks, one or more new sources/sinks being added to the domain and/or one or more of the sources/sinks being removed from the domain. For example, by way of reference to a data center, the air flow at one or more of the vents or perforated tiles may be increased or decreased and/or one or more of the vents or perforated tiles may be relocated or removed. As will be described in detail below, advantageously, as long as the domain/finite element mesh does not change, then these elements can be recycled in subsequent iterations of the methodology saving on processing time.

[0034] Each time the above-described steps are repeated, updated air flow measurements from the air flow sources and sinks can be acquired (if available) therefore making the instant techniques sensitive to changing conditions within the space. By way of example only, steps 210-214 can be repeated periodically (for example at a pre-determined time interval) to acquire updated air flow measurement data. The pre-determined time interval can be set, for example, based on a frequency at which updated measurement data is available and/or a frequency with which changes occur in the space.

[0035] In step 216, a cooling capacity is determined for each air flow source. For example, with reference to data center modeling, the cooling capacity of each vent or perforated tile

can be determined. An exemplary process for determining cooling capacity is described in detail below.

[0036] As highlighted above, in one exemplary embodiment, potential flow theory is employed assuming constant (temperature independent) air density, free slipping over boundaries and that viscous forces can be neglected. For a general description of potential flow theory see, for example, L.D. Landau et al., "Fluid Mechanics," Pergamon Press (1959), the contents of which are incorporated by reference herein.

[0037] As the air velocity $v = (v_x, v_y, v_z)$ is assumed to be irrotational, that is, curl $v = 0$, the velocity can be taken to be the gradient of a scalar function ϕ. This function ϕ is called the "velocity (or air flow) potential" and it satisfies the Poisson equation. In other words, the (air) velocity field corresponds to a solution of:

$$\left(\frac{\partial^2 \phi}{\partial x^2}\right) + \left(\frac{\partial^2 \phi}{\partial y^2}\right) + \left(\frac{\partial^2 \phi}{\partial z^2}\right) = f \qquad (1)$$

$$v_x = \frac{\partial \phi}{\partial x} \quad v_y = \frac{\partial \phi}{\partial y} \quad v_z = \frac{\partial \phi}{\partial z} \qquad (2)$$

with appropriate boundary conditions. Here, f represents flow sources or sinks and v_x, v_y and v_z are the velocity components in the x, y and z directions, respectively. It is notable that other more comprehensive partial differential equations (PDEs) can be used in accordance with the present techniques, which may include turbulence models and dissipation.

[0038] In order to provide boundary conditions for the above problem one could, for example, model vents or perforated tiles (or the output of ACUs) as "sources" ($\frac{\partial \phi}{\partial z}$ = - (measured) output velocity from a perforated tile), the returns to the ACUs as "sinks" ($\varphi = 0$), while the racks are sinks ($\frac{\partial \phi}{\partial x}$ = (measured) inlet rack flow) and sources ($\frac{\partial \phi}{\partial x}$ = -(measured) outlet rack flow) at the same time. As described above, the sources and sinks can vary depending on the domain (e.g., when the domain includes the sub-floor plenum the perforated tiles and ACUs can be considered sinks and sources, respectively; on the other hand when the domain includes the raised floor the perforated tiles and ACUs can be considered sources and sinks, respectively). Another alternative is to model the sources and sinks via a non-zero right-hand side f in Equation 1. If the sources or sinks are located on boundaries of the solution domain of the PDE, the former approach is appropriate (as it corresponds to the

specification of Neumann boundary conditions). For sources or sinks that are located inside the solution domain (that is, not on boundaries), modeling via a non-zero right-hand side f in Equation 1 would be more suitable.

[0039] A finite element solver is implemented herein to calculate the air flow potential. For a standard reference on the finite element method, see T.J.R. Hughes, "The Finite Element Method: Linear Static and Dynamic Finite Element Analysis," Chapter 1, Dover Publications (2000) (Originally published by Prentice-Hall, 1987), the contents of which are incorporated by reference herein. One exemplary implementation of the present techniques was done in the C programming language, following some analogous finite element implementations in Matlab by J. Alberty et al., "Remarks Around 50 Lines of Matlab: Short Finite Element Implementation," Numerical Algorithms 20, pp. 117-137 (1999) (hereinafter "Alberty"), the contents of which are incorporated by reference herein.

[0040] The solver requires specification of a mesh, which consists of a set of nodes within a specified domain, as well as triangles (for two-dimensional domains) or tetrahedra (for three-dimensional domains) connecting these nodes. As highlighted above, the triangles (or tetrahedra) are also referred to herein as elements of the mesh. The nodes correspond to (x, y) coordinates in a two-dimensional domain (or (x, y, z) coordinates in a three-dimensional domain). An example of a finite element mesh is shown illustrated in FIG. 3. Namely, FIG. 3 is a diagram illustrating an exemplary finite element mesh 300 that includes a set of nine nodes and eight elements in a two-dimensional domain. A key to each of the eight elements E and the corresponding nodes is provided to the right of the mesh.

[0041] An approximate solution,

$$\phi \approx \sum_{i=1}^{N} \hat{\phi}_i \Psi_i \qquad (3)$$

is sought to Equation 1 as a linear combination of N basis functions $\Psi_i, i = 1, \ldots, N$. Letting $\vec{x}_j$ denote the j-th node in the mesh, the basis functions are chosen so that $\Psi_i(\vec{x}_j) = 1$ if $i = j$ and $\Psi_i(\vec{x}_j) = 0$ otherwise. This is typical of finite element approximations and has the advantage that the coefficient $\hat{\phi}_i$ in the linear combination of Equation 3 also corresponds to the approximation of the solution at the i-th node. As in Alberty, piecewise linear basis functions can be used to approximate the solution of Equation 1. Again, this is a standard

choice in finite element approximations. Upon application of a Galerkin finite element discretization to Equation 1 with suitable boundary conditions one obtains a system of linear equations,

$$A\hat{\phi} = b. \tag{4}$$

The process for applying a Galerkin finite element discretization to Equation 1 to result in the system of Equations 4 would be apparent to one of skill in the art, and thus is not described further herein. The solution $\hat{\phi} = (\hat{\phi}_1, \ldots, \hat{\phi}_N)$ of Equation 4 gives an approximate solution to the air flow potential ϕ at the nodes of the finite element mesh. Since the interest is in obtaining the gradient of the potential, as it defines the velocity field in Equations 2, once the linear system of Equations 4 is solved, a numerical approximation to the gradient of ϕ can be obtained from the linear combination of Equation 3 and the solution $\hat{\phi}$ of the system of Equations 4. This provides the air flow field, as defined by Equations 2. However, due to the choice of basis functions, it is not meaningful to obtain an approximation to the gradient at the mesh nodes. Instead, an approximation is valid at points inside each element. The center of each element can be chosen as a convenient and standard coordinate point, i.e., at which to approximate the velocity field. Thermal zones can then be defined from trajectories of the air flow, as described below.

[0042] A two-dimensional implementation of the above techniques to calculate the air flow potential for the sub-floor plenum area of a large (e.g., greater than 50,000 square feet) data center will now be described. In the following, the focus is on "plenum" thermal zones but the same principles can be applied to "above plenum" or the total raised floor data center. FIG. 4 is a diagram illustrating a graphical representation 400 of a data center, which includes a plurality of ACUs 402 and perforated tiles 404. Such a graphical representation can be created, for example, using MMT Client, see below. Certain areas in the data center, such as area 406 are excluded from the calculations for example since area 406 is separated from the plenum. The domain boundary 408 defines the physical dimensions of the entire plenum.

[0043] As described above, with a domain defined by the sub-floor plenum the ACUs can be the air flow sources while the perforated tiles can be the air flow sinks applied as Neumann boundary conditions. In this particular embodiment, the right hand side of Equation 1 is set to zero. As highlighted above, air flow measurement data for the ACUs and the perforated

tiles can be obtained using MMT. Suitable techniques to obtain the required input data for both the ACUs and perforated tiles are also described, for example, in H.F. Hamann, et al., "Methods and Techniques for Measuring and Improving Data Center Best Practices" IEEE Proceedings of the ITherm 2008 Conference, Orlando, Florida, pp. 1146-1152 (May 2008), the contents of which are incorporated by reference herein.

[0044] An enlarged view of a portion 500 of graphical representation 400 shown in FIG. 5 illustrates that a mesh has been created. The nodes are connected in this two-dimensional model with triangles. The sources (e.g., ACUs) and sinks (e.g., perforated tiles) are shown labeled in FIG. 5.

[0045] FIG. 6 is a diagram illustrating air flow potential ϕ as well as the corresponding air flow velocity vectors that result from performing the above-described calculations. The same portion 500 of graphical representation 400 is shown in FIG. 6. The darker areas correspond to a high air flow potential while the lighter shaded areas correspond to a lower air flow potential. As shown in FIG. 6, some of the ACUs have been turned off and thus no boundaries were applied. The underlying grid indicates the tile grid, which is typically used in data centers.

[0046] Once the air flow velocity field has been calculated (according to Equation 2, above), the air flow from/to each area of the data center is traced back to the originating/returning ACU (the air flow velocity vectors collectively indicate air flow patterns, which is why the velocity field is used to define the thermal zones (which are given by these air flow patterns)). An exemplary methodology for tracing the air flow in a data center is described, for example, in conjunction with the description of FIG. 12 below. In this particular example, the air flow is traced throughout the sub-floor plenum, but could also be done in the same manner for the raised floor as well. The trace connects a specific ACU with its corresponding thermal zone.

[0047] These traces (also referred to herein as "air flow trajectories"), which are paths that individual particles follow, as well as the corresponding thermal zones, are shown in FIG. 7. Again, the same portion 500 of graphical representation 400 is shown in FIG. 7. In FIG. 7, the thermal zones are outlined (by a zone boundary). Those ACUs with a hatched pattern are turned off. In case the conditions are changing, for example, a perforated tile gets removed or different tile types are deployed or an ACU gets turned off, the calculations can be repeated and the data center plenum re-zoned accordingly. Different types of tiles have

different levels of perforation which is how one can control how much air flow can be pushed through the tile for a given pressure gradient. Commonly, reference is made to the "level" of openings in the tile. So, for example, there are 25% open tiles up to 50% open tiles. A 25% open tile has a higher resistance than the 50% open tile. The resistance or impedance R is a property of the tile openness and some other factors. If the pressure differential p is known then the air flow f can be calculated by:

$$p=R \times f^2$$

[0048] At this point it is worth noting that, as long as such changes do not involve modifications to the finite element mesh, repeating the calculations will require less computing time since the nodes and elements do not have to be regenerated each time. Furthermore, if a direct solver for linear systems is used to solve Equations 4, savings in computation time are also possible as long as the matrix A in Equations 4 remains unchanged (which is the case if the conditions changing correspond only to modifications in the measured flow at the sources and sinks, as this results only in a modified right-hand side b in Equations 4). As a direct solver typically employs two phases, numerical factorization of the coefficient matrix followed by the solution of the system with the factored matrix, the numerical factorization of the coefficient matrix need only be done once, as long as the matrix remains unchanged. The numerical factorization is the most time consuming of the two phases, so doing the factorization only when strictly needed can result in considerable savings in computational time.

[0049] The present techniques may include the exploitation of the superposition principle, which may provide ways to faster solve the respective air flow patterns for varying conditions by avoiding redundant calculations. Specifically, Equation 1 being solved for the potential ϕ can be re-written as $\nabla\cdot(\nabla\phi)=f$, where $\nabla\cdot$ is the divergence operator. One can use the principle of superposition and sum two solutions ϕ_1 and ϕ_2 of Equation 1 (or, generally, any number of solutions) to obtain a third solution $\phi_3=\phi_1+\phi_2$ as long as ϕ_1 and ϕ_2 are solutions to Equation 1 for the same domain (geometry). That is, say ϕ_1 solves $\nabla\cdot(\nabla\phi_1)=f_1$ and ϕ_2 solves $\nabla\cdot(\nabla\phi_2)=f_2$, then $\phi_3=\phi_1+\phi_2$ solves $\nabla\cdot(\nabla\phi_3)=f_1+f_2$. For example, say ϕ_1 is a solution obtained with only ACU1 on (at a given fan speed setting), while all the other ACUs were off, and ϕ_2 is a solution obtained with only ACU2 on (at a given fan speed setting), while all the other ACUs were off. The velocity field for the scenario with only ACU1 on is $v_1=\nabla\phi_1$ and the

velocity field for the scenario with only ACU2 on is $v_2 = \nabla\phi_2$. Then $v_3 = v_1 + v_2$ corresponds to a velocity field for the scenario with ACU1 and ACU2 on (at the corresponding fan speeds for which the original scenarios were obtained), while all other ACUs are off.

[0050] In one exemplary embodiment a graphical software (e.g., MMT client) is used to provide a graphical representation of the space, e.g., data center, define the domains, edit sinks and sources, feed sensor data (if available) to define sinks and sources, initialize the calculations, postprocess and visualize the thermal zones. MMT client is a software application, which allows graphically displaying the data center layout and also visualizing the air flow trajectories, the zones and the air flow vectors.

[0051] See, for example, FIG. 8 which is a diagram illustrating exemplary graphical interface 800 (as displayed for example on a video display, such as a computer monitor, see below) containing a graphical representation 801 of a data center. For illustrative purposes, the structures within the data center, such as server racks and perforated tiles, are labeled in graphical representation 801 (and may or may not be labeled in an actual implementation of the process). The domain of the plenum is defined, for example by boundary 806, as a polygon of x- and y-points. Here two domains have been defined, an outer and an inner (the inner domain is not visible in this depiction). The outer domain circumferences the plenum of the data center while the inner covers a control room which is separate. Dirichlet boundary conditions have also been defined (where the potential is set to a certain value - in most cases zero).

[0052] As highlighted above, the air flow at one or more of the sources/sinks can change. Further, sources/sinks can be added to and/or removed from the domain. FIG. 9 is a diagram illustrating how interface 800 (of FIG. 8) can be graphically edited to reflect such changes in the domain. In this particular example the air flow is specified for a perforated tile. The perforated tile can be interactively moved, duplicated and tile properties can be readily manipulated (type, air flow impedance etc.). In one exemplary embodiment, real-time sensors located for example in the plenum can be used to update the model input. Specifically, by measuring the pressure in the plenum and in combination with air flow impedance of the vent or the perforated tile, the air flow can be automatically calculated and then applied to the zoning model. The perforated air flow tile impedance R [Pa/cfm^2] is given by the following equation,

$$R = \frac{1}{2} \cdot \frac{\rho}{A^2} \cdot K$$

with ρ as the density of air, A as the area of the tile and K as a loss coefficient. The pressure difference the air flow are related as follows,

$$\Delta p = R \cdot f^2_{airflow}.$$

The pressure differential is re-measured with real-time sensors and use $\Delta p = R \cdot f^2_{airflow}$ to calculate air flow through each tile. As discussed the air flow is applied as a boundary.

[0053] Once the model has been defined and set up (i.e., a graphical representation has been provided, all domains have been defined, etc.) the thermal zones can be modeled (as described above) either based on user input or triggered by measured change in the data center using a sensor network (which, for example, can detect air flow changes and/or the addition/removal of sources/sinks from the domain, see above). FIG. 10 is a diagram illustrating the visualization of thermal zones on interface 800 (of FIG. 8). In the exemplary embodiment shown in FIG. 10, gray shading is used to differentiate the energy efficiency of the various thermal zones, with the shading used for example in thermal zone 1002 denoting the greatest efficiency. As highlighted above, efficiency can be characterized by "Utilization" or better by "COP." If one increases "heat removed" at a given nominator COP and utilization increases and thus efficiency increases (i.e., utilization (UT) = heat removed / nominal heat load removal capacity) or coefficient of performance (COP) (COP=heat removed / power consumption for ACU fans). Bar charts now present at the bottom of interface 800 illustrate the utilization and COP. Lines 1004 and 1006 above the bar charts correspond to discharge and return temperatures of the ACUs, respectively, which can be obtained either from static data (which can be edited in MMT client) or from real-time data (being fed into the MMT client from real-time sensors, e.g., so as to update conditions in the data center as described above). For illustrative purposes one of the thermal zones (having a utilization of 54% and a COP of nine) is shown linked by arrow 1008 to the corresponding bar chart.

[0054] Finally, it is notable that the present techniques can also be used to determine a respective cooling capacity from each vent or perforated tile. Namely, in conjunction with a temperature model (as described in Hamann) the vent or perforated tile discharge temperatures T_D can be calculated (i.e., once a velocity field $\vec{v} = (v_x, v_y, v_z)$ is obtained, it is

used in the energy equation $\rho c_p \underline{v}\, grad(T) + div(kg\, rad(T)) = 0$ with the temperature prescribed as the boundaries (e.g., at the inlet and outlet of the servers) in order to solve for the temperature distribution). Thus, the temperature distribution can be calculated (for example in two-dimensions) as a function of an x and y coordinate T(x,y). By knowing where (xt,yt) the perforated tiles are one can get the tile discharge temperature Td=T(xt,yt).

[0055] In combination with the air flow velocity or total air flow and an allowable inlet temperature T_{inlet} for the server, the cooling power per tile can be determined for each vent/perforated tile by $P_{cool} \approx (T_{inlet} - T_D) \cdot flow / 3140[cfmK/kW]$. In case the velocity vectors have been calculated (rather than measured and used as a boundary as in this embodiment), the flow can be obtained by integrating the perpendicular velocity vector component (most often v_z) over the area of the tile or vent $flow = \int v_\perp dA$. An example, of this feature is shown in FIG. 11. Specifically, FIG. 11 is a schematic diagram 1100 illustrating cooling capacity in a data center. In FIG. 11, the layout representation has been overlayed as pseudo three-dimensional image with a three-dimensional bar chart, wherein the height of the bars show the cooling capacity (i.e., cooling power).

[0056] FIG. 12 is a diagram illustrating exemplary methodology 1200 for tracing air flow velocity vectors to an air flow source, such as an ACU. Methodology 1200 represents an exemplary process for performing step 214 of methodology 200 (of FIG. 2) described above. In step 1202, the process begins at a given location (which is also considered an initial location if this is the first iteration of the process), thus a given (e.g., an initial) location is selected. According to an exemplary embodiment, when the space being modeled is a data center, each location described in accordance with this process can correspond to the location of a given tile (any tile, not just a perforated tile) in the data center raised floor. In that instance, the initial location can be any tile. The process can begin with that initial tile (location) and then loop through all tiles in the data center. In step 1204, the velocity vector for the given location (which is the initial location in the first iteration of the process) is referenced (i.e., the velocity vectors throughout the data center have already been determined, see for example step 212 of methodology 200, described above). As described above, the air flow velocity vectors throughout a space, i.e., data center, can be determined using finite element techniques (i.e., within a plurality of sub-domains) based on air flow data obtained from sources/sinks. If, however, the velocity vector for the given location has not already been determined, it can be extracted using standard interpolation techniques. For example,

when calculating the trajectory a location is eventually reached that does not have a specifically calculated air flow vector. But of course there are locations close by that have a calculated air flow vector. Several different approaches have been tested for such a scenario. One approach is to use the closest velocity vector. Another approach is to use a distance weighted average of a plurality (e.g., of nine) neighboring velocity vectors. In general, however, any interpolation technique can be used.

[0057] In step 1206, a new next location is determined, i.e., calculated. Using the above example of a data center tile as location point, in this step the next tile is selected. In a two-dimensional domain, the next location xt and yt can be selected as follows:

$$xt(steps) = (xt(steps-1) + vx*stepsize*n); \text{ and}$$

$$yt(steps) = (yt(steps-1)+vy*stepsize*n),$$

wherein the magnitude of the velocity vector $v = sqrt(vx^2+vy^2)$ and $n = 0.2/v$. $v = sqrt(vx^2+vy^2)$....$n = 0.2 / v$ is a parameter which controls the stepsize in relationship to vx and vy and prevents making too small of a step if the velocity vectors are small. In other words, the variable n is introduced to make sure that, if velocities are very small, one still moves a sizeable step from one location to the next so that the methodology is not too slow. Stepsize can be chosen by the user.

[0058] In step 1208, a determination is made as to whether the new location falls within an area of an airflow source, e.g., an ACU. If the trajectory intersects with an ACU, then in step 1210, the previous location is assigned to that ACU. For example, if tile 1 is the initial (previous) location and tile 2 is the new location, and tile 2 falls within an area of ACU 1, then tile 1 would be assigned to ACU 1. The process is then repeated (for n number of locations, e.g., tiles, throughout the space, e.g., data center) by referencing the velocity vector for the new location, e.g., tile 2, (step 1204), determining another new location, e.g., of a tile 3, (step 1206) and so on. The ACU location is known for example from the graphical representation in the MMT client. That representation includes the x,y coordinates of the ACU as well as the width, length and height (i.e., defining the area) of the ACU. Once xt and yt (see step 1206) are within ACU area, another new location is determined, and so on.

[0059] On the other hand, if the new location (e.g., tile 2 using the above example) does not fall within an area of an ACU, then a determination can be made in step 1212 as to whether or not too many steps have been made, i.e., there is a maximum number of steps in case the

trajectory does not end up at an ACU. Another way to look at it is there may be a limit imposed on the number of times steps 1204 and 1206 can be repeated without having a location (e.g., tile) fall within the area of an air flow source. If in fact too many steps have been made (i.e., the limit has been reached), then in step 1214, the previous location, e.g., tile 1, is designated as a "no zone," meaning that location is not associated with a thermal zone. The process is then repeated by referencing the velocity vector for the new location, e.g., tile 2, (step 1204), determining another new location, e.g., of a tile 3, (step 1206) and so on for n locations throughout the space. On the other hand, if the maximum number of steps has not been exceeded (i.e., the limit has not been reached) (and the new location, e.g., tile 2, does not fall within the area of an ACU (step 1208)) then the process beginning again at step 1204 is repeated, i.e., by referencing the velocity vector for the new location, e.g., tile 2, (step 1204), determining another new location, e.g., of a tile 3, (step 1206) and so on. Methodology 1200 is repeated for each of n locations, i.e., tiles, in the space, i.e., data center. In this manner, each velocity vector is traced to a particular air flow source (e.g., ACU) or designated as not being associated with a particular thermal zone. Exemplary code for tracing velocity vectors to a particular air flow source and thereby defining thermals zones in a space is provided below.

[0060] Turning now to FIG. 13, a block diagram is shown of an apparatus 1300 for modeling thermal zones in a space, e.g., within a room, such as data center 100 (FIG. 1), in accordance with one embodiment of the present invention. It should be understood that apparatus 1300 represents one embodiment for implementing methodology 200 of FIG. 2.

[0061] Apparatus 1300 comprises a computer system 1310 and removable media 1350. Computer system 1310 comprises a processor device 1320, a network interface 1325, a memory 1330, a media interface 1335 and an optional display 1340 (for displaying, e.g., graphical interface 800 of FIG. 8, described above). Network interface 1325 allows computer system 1310 to connect to a network, while media interface 1335 allows computer system 1310 to interact with media, such as a hard drive or removable media 1350.

[0062] As is known in the art, the methods and apparatus discussed herein may be distributed as an article of manufacture that itself comprises a machine-readable medium containing one or more programs which when executed implement embodiments of the present invention. For instance, the machine-readable medium may contain a program configured to provide a graphical representation of the space; define at least one domain in the space for modeling; create a mesh in the domain by sub-dividing the domain into a set of discrete sub-domains

that interconnect a plurality of nodes; identify air flow sources and sinks in the domain; obtain air flow measurements from one or more of the air flow sources and sinks; determine an air flow velocity vector at a center of each sub-domain using the air flow measurement obtained from the air flow sources and sinks; and trace each velocity vector to one of the air flow sources, wherein a combination of the traces to a given one of the air flow sources represents a thermal zone in the space.

[0063] The machine-readable medium may be a recordable medium (e.g., floppy disks, hard drive, optical disks such as removable media 1350, or memory cards) or may be a transmission medium (e.g., a network comprising fiber-optics, the world-wide web, cables, or a wireless channel using time-division multiple access, code-division multiple access, or other radio-frequency channel). Any medium known or developed that can store information suitable for use with a computer system may be used.

[0064] Processor device 1320 can be configured to implement the methods, steps, and functions disclosed herein. The memory 1330 could be distributed or local and the processor 1320 could be distributed or singular. The memory 1330 could be implemented as an electrical, magnetic or optical memory, or any combination of these or other types of storage devices. Moreover, the term "memory" should be construed broadly enough to encompass any information able to be read from, or written to, an address in the addressable space accessed by processor device 1320. With this definition, information on a network, accessible through network interface 1325, is still within memory 1330 because the processor device 1320 can retrieve the information from the network. It should be noted that each distributed processor that makes up processor device 1320 generally contains its own addressable memory space. It should also be noted that some or all of computer system 1310 can be incorporated into an application-specific or general-use integrated circuit.

[0065] Optional video display 1340 is any type of video display suitable for interacting with a human user of apparatus 1300. Generally, video display 1340 is a computer monitor or other similar video display.

[0066] The following is example code in a pv-wave programming language for tracing velocity vectors to a particular air flow source and thereby defining thermals zones in a space.

```
; {{{ Pdata
Function mmts_Pdata,x,y,area=area,dir=dir
 pdata=fltarr(6,n_elements(x))    &    pdata(0,*)=x    &    pdata(1,*)=y    &
area=4.0*poly_area(pdata(0,*),pdata(1,*))
```

```
 dir=strarr(n_elements(pdata(0,*)))
 for i=0,n_elements(pdata(0,*))-2 do begin
  x0=nint(pdata(0,i)*2.0)        &        y0=nint(pdata(1,i)*2.0)        &
x1=nint(pdata(0,i+1)*2.0) & y1=nint(pdata(1,i+1)*2.0)
  case 1 of & x1 GT x0: dir(i+1)='S' & y1 GT y0: dir(i+1)='E' & x0 GT x1:
dir(i+1)='N' & y0 GT y1: dir(i+1)='W'
  else: print,'problem in domain section' & endcase
 endfor
 dir(0)=dir(i) & n=n_elements(pdata(0,*))
 for i=0,n-2 do begin
  corner=strcompress(dir(i)+dir(i+1))
  case 1 of
   corner EQ 'NE' or corner EQ 'EN':begin & pdata(2,i)=pdata(0,i)-0.25 &
pdata(3,i)=pdata(1,i)-0.25       &       pdata(4,i)=pdata(0,i)+0.25       &
pdata(5,i)=pdata(1,i)+0.25 & end
   corner EQ 'ES' or corner EQ 'SE':begin & pdata(2,i)=pdata(0,i)-0.25 &
pdata(3,i)=pdata(1,i)+0.25       &       pdata(4,i)=pdata(0,i)+0.25       &
pdata(5,i)=pdata(1,i)-0.25 & end
   corner EQ 'SW' or corner EQ 'WS':begin & pdata(2,i)=pdata(0,i)+0.25 &
pdata(3,i)=pdata(1,i)+0.25       &       pdata(4,i)=pdata(0,i)-0.25       &
pdata(5,i)=pdata(1,i)-0.25 & end
   corner EQ 'WN' or corner EQ 'NW':begin & pdata(2,i)=pdata(0,i)+0.25 &
pdata(3,i)=pdata(1,i)-0.25       &       pdata(4,i)=pdata(0,i)-0.25       &
pdata(5,i)=pdata(1,i)+0.25 & end
  else: print,'problem in domain section'
  endcase
 endfor
 pdata(*,i)=pdata(*,0)
 return,pdata
end
;}}}

;{{{ DomNodes
Pro mmts_DomNodes,pdata,r,dom_nodes=dom_nodes,id=id,od=od
 dxy=size(dom_nodes)
 case 1 of
  keyword_set(id):  v=-1 & keyword_set(od): v=0
 else: print,'od/id keyword missing'
 endcase
 if                              keyword_set(id)                              then
dom_nodes(polyfillv(r*pdata(2,*)+1,r*pdata(3,*)+1,dxy(1),dxy(2)))=v     else
dom_nodes(polyfillv(r*pdata(4,*)+1,r*pdata(5,*)+1,dxy(1),dxy(2)))=v
endif
 ;}}}
```

```
Function
mmts_CalculatePlenumTemp,gflow,gtemp,acu,xl,yl,points=points,gcool=gcool


 ai1=where(gflow GT min(gflow)) & ai2=where(acu(13,*) EQ 'YES')
 points=fltarr(3,n_elements(ai1)+n_elements(ai2))


 x=(float(acu(1,ai2))+float(acu(4,ai2))/2.0)/2.0                    &
y=(float(acu(2,ai2))+float(acu(5,ai2))/2.0)/2.0
 points(0,n_elements(ai1):n_elements(ai1)+n_elements(ai2)-1)=x
 points(1,n_elements(ai1):n_elements(ai1)+n_elements(ai2)-1)=y
 points(2,n_elements(ai1):n_elements(ai1)+n_elements(ai2)-
1)=float(acu(11,ai2))


 x=ai1-(great_int(ai1)/xl)*xl & y=great_int(ai1)/xl
 points(0,0:n_elements(ai1)-1)=x+0.5 & points(1,0:n_elements(ai1)-1)=y+0.5
 for                    i=0,n_elements(ai1)-1                       do
points(2,i)=avg(gtemp(3*x(i):3*x(i)+2,3*y(i):3*y(i)+2,0:0))*1.8+32.0


 out=mmts_IWDInt3D(points,xl,yl)
 gcool=fltarr(xl,yl) & gcool(*,*)=1000
 gcool(x,y)=gflow(x,y)*(90.0-out(x,y))/3140.0
 gcool(where(gcool EQ 1000))=min(gcool)-0.2
 return,out
end
;}}}
;{{{FEAutoMesherNodes2D
Pro
mmts_FEAutoMesherNodes2D,name,domains,r,glayout,obstruct=obstruct,dom_nodes
=dom_nodes,nodes=nodes,total_area=total_area,Show_nodes=Show_nodes,Show_bou
ndary=Show_boundary
 r=float(r) & total_area=0
 dd=where(domains(0,*) EQ name) & ds=sort(domains(4,dd))
 if ds(0) EQ -1 then print,'domain problem in '+name
 for d=0,n_elements(ds)-1 do begin
  x=float(strsplit(domains(2,ds(d)),';'))                           &
y=float(strsplit(domains(3,ds(d)),';'))
  pdata=mmts_Pdata(x,y,area=area)
  if domains(4,ds(d)) EQ 'inner' then begin & total_area=total_area+area &
mmts_DomNodes,pdata,r,dom_nodes=dom_nodes,/id
  endif      else      begin      &      total_area=total_area-area      &
mmts_DomNodes,pdata,r,dom_nodes=dom_nodes,/od & endelse
  if        keyword_set(Show_boundary)        then        begin        &
plots,pdata(0:1,*),color=3,thick=3                                      &
```

```
plots,pdata(2:3,*),color=4,thick=3,Linestyle=1                                    &
plots,pdata(4:5,*),color=5,thick=3,Linestyle=1 & endif

 endfor


 if keyword_set(obstruct) then begin

  for  i=0,n_elements(obstruct)-1  do  begin  &  ai=where(glayout(0,*)  EQ
obstruct(i))

   if ai(0) GT -1 then begin

    for n=0,n_elements(ai)-1 do begin

     x=float(glayout(1,ai(n)))*r     &     y=float(glayout(2,ai(n)))*r     &
dx=float(glayout(4,ai(n)))*r & dy=float(glayout(5,ai(n)))*r

     dom_nodes(nint(x)+1:nint(x)+nint(dx)-1,nint(y)+1:nint(y)+nint(dy)-1)=0

     if                    keyword_set(Show_boundary)                    then
plots,[x/r,x/r+dx/r,x/r+dx/r,x/r,x/r],[y/r,y/r,y/r+dy/r,y/r+dy/r,y/r],color
=3

    endfor

   endif

  endfor

 endif


 nodes=strarr(3,long(n_elements(where(dom_nodes(*,*) EQ -1))))

 q=long(0)

 for      y=0,n_elements(dom_nodes(0,*))-1      do      begin      &      for
x=0,n_elements(dom_nodes(*,0))-1 do begin

  if dom_nodes(x,y) EQ -1 then begin

   if                      keyword_set(Show_nodes)                      then
oplot,[x/r],[y/r],psym=4,color=2,symsize=0.25

   nodes(0,q)=q+1  &  nodes(1,q)=string(x/r)  &  nodes(2,q)=string(y/r)    &
dom_nodes(x,y)=q+1 & q=q+1

  endif

 endfor & endfor

 end

;}}}

;{{{FEAutoMesherTriangles2D

Pro
mmts_FEAutoMesherTriangles2D,r,nodes,dom_nodes,dom_triangles=dom_triangles,
triangles=triangles,Show_triangles=Show_triangles

 d=1.0/r  & q=long(0) & dxy=size(dom_triangles) & xl=dxy(1)/2 & yl=dxy(2)/2

 triangles=strarr(4,long(100000)) & q=long(0)

 for i1=long(0),n_elements(nodes(0,*))-1 do begin

  x=float(nodes(1,i1)) & y=float(nodes(2,i1))

  if x LT float(xl) and y LT float(yl) then begin

   p0=dom_nodes(nint(x*r),nint(y*r))                                          &
p1=dom_nodes(nint((x+d)*r),nint(y*r))
```

```
   p2=dom_nodes(nint((x+d)*r),nint((y+d)*r))                                    &
p3=dom_nodes(nint(x*r),nint((y+d)*r))
   if p0 GT 0 and p1 GT 0 and p2 GT 0 and p3 GT 0 then begin
    if p0 GT 0 and p1 GT 0 and p3 GT 0 then begin
     if                  keyword_set(Show_triangles)                  then
oplot,[x,x+d,x,x],[y,y,y+d,y],color=2,thick=0.01
     triangles(0,q)=q+1   &   triangles(1,q)=p0   &   triangles(2,q)=p1   &
triangles(3,q)=p3 & q=q+1

dom_triangles(nint(x*r),nint(y*r))=dom_triangles(nint(x*r),nint(y*r))+strin
g(q)
    endif
    if p1 GT 0 and p2 GT 0 and p3 GT 0 then begin
     if                  keyword_set(Show_triangles)                  then
oplot,[x+d,x+d,x,x+d],[y,y+d,y+d,y],color=2,thick=0.01
     triangles(0,q)=q+1   &   triangles(1,q)=p1   &   triangles(2,q)=p2   &
triangles(3,q)=p3 & q=q+1

dom_triangles(nint(x*r),nint(y*r))=dom_triangles(nint(x*r),nint(y*r))+','+s
tring(q)
    endif
   endif
  endif
 endfor
 dom_triangles=strcompress(dom_triangles,/remove_all)                         &
triangles=triangles(*,0:q-1)
end
;}}}
;{{{FESourcesSinks
Pro
mmts_FESourcesSinks,acu,acuMetrics,gflow,dom_triangles,r,sources_sinks=sour
ces_sinks,Show_sources=Show_sources,Show_sinks=Show_sinks
 r=float(r)  &  sources_sinks=strarr(3,100000)   &  q=0  &  sum_sources=0  &
sum_sinks=0 & acu_on=where(acu(13,*) EQ 'YES')
 for i=0,n_elements(acu_on)-1 do begin
  x=float(acu(1,acu_on(i)))   /2.0   &   y=float(acu(2,acu_on(i)))/2.0   &
dx=float(acu(4,acu_on(i)))/2.0 & dy=float(acu(5,acu_on(i)))/2.0
  for     ix=nint(x*2),nint((x+dx)*2)-1     do     begin     &     for
iy=nint(y*2),nint((y+dy)*2)-1 do begin
   tr=strsplit(dom_triangles(ix,iy),',')
   if n_elements(tr) EQ 2 then begin
    sources_sinks(0,q:q+1)=tr(*)                                              &
sources_sinks(1,q:q+1)=acuMetrics(1,acu_on(i))/dx/dy/(-8.0)                   &
sources_sinks(2,q:q+1)=' '+acuMetrics(0,acu_on(i))
    q=q+2                                                                     &
sum_sources=sum_sources+2.0*float(acuMetrics(1,acu_on(i)))/dx/dy/(-8)
```

```
    if keyword_set(Show_sources) then begin

oplot,[ix/2.0,ix/2.0+1.0/float(r),ix/2.0+1.0/float(r),ix/2.0,ix/2.0],[iy/2.
0,iy/2.0,iy/2.0+1.0/float(r),iy/2.0+1.0/float(r),iy/2.0],color=7

oplot,[ix/2.0+1.0/float(r),ix/2.0],[iy/2.0,iy/2.0+1.0/float(r)],color=7
    endif
   endif
  endfor & endfor
 endfor
 q1=q & dxy=size(gflow)
 for ix=0,2*dxy(1)-1 do begin & for iy=0,2*dxy(2)-1 do begin
  if gflow(great_int(ix/2),great_int(iy/2)) GT min(gflow) then begin
   tr=strsplit(dom_triangles(ix,iy),',')
   if n_elements(tr) EQ 2 then begin
    sources_sinks(0,q:q+1)=tr(*)                                          &
sources_sinks(1,q:q+1)=float(gflow(great_int(ix/2),great_int(iy/2)))/8.0  &
sources_sinks(2,q:q+1)='      PERF'
    q=q+2                                                                 &
sum_sinks=sum_sinks+2.0*float(gflow(great_int(ix/2),great_int(iy/2)))/8.0
    if keyword_set(Show_sinks) then begin

oplot,[ix/2.0,ix/2.0+1.0/float(r),ix/2.0+1.0/float(r),ix/2.0,ix/2.0],[iy/2.
0,iy/2.0,iy/2.0+1.0/float(r),iy/2.0+1.0/float(r),iy/2.0],color=9

oplot,[ix/2.0+1.0/float(r),ix/2.0],[iy/2.0,iy/2.0+1.0/float(r)],color=9
    endif
   endif
  endif
 endfor & endfor
 sources_sinks=sources_sinks(*,0:q-1)
 sources_sinks(1,0:q1-1)=double(sources_sinks(1,0:q1-1))/
sum(double(sources_sinks(1,0:q1-1)))*(-1)
 sources_sinks(1,q1:q-1)=double(sources_sinks(1,q1:q-1))/
sum(double(sources_sinks(1,q1:q-1)))
end
;}}}
;{{{FESolverPDE
Pro
mmts_FESolver,PDE_Path,DC_path,triangles,nodes,dirichlet,neumann,sources_si
nks,index_type,name_pot,name_vel

 name_nodes='~nodes.txt'                                                  &
status=dc_write_free(strcompress(PDE_path+name_nodes,/remove_all),nodes,/co
lumn,Delim=',')
```

```
 name_triangles='~triangles.txt'                                          &
status=dc_write_free(strcompress(PDE_path+name_triangles,/remove_all),trian
gles,/column,Delim=',')

 name_dirichlet='~dirichlet.txt'                                          &
status=dc_write_free(strcompress(PDE_path+name_dirichlet,/remove_all),diric
hlet,/column,Delim=',')

 name_neumann='none'

                                                  ; today no Neumann file

 name_sources_sinks='~sources_sink.txt'                                   &
status=dc_write_free(strcompress(PDE_path+name_sources_sinks,/remove_all),s
ources_sinks,/column,Delim=',')

 name_input='~input.txt' & name_pot='~pot.txt' & name_vel='~vel.txt'

 n_triang=n_elements(triangles(0,*)) & n_nodes=n_elements(nodes(0,*))

 n_dirichlet=1              &              n_neumann=0              &
n_sources_sinks=n_elements(sources_sinks(0,*))    &    index_type=1    &
input_dat=strarr(13)

 input_dat(0)='number              of              elements              =
'+strcompress(string(n_triang),/remove_all)  &  input_dat(1)='elements  file
name = '+name_triangles

 input_dat(2)='number of nodes = '+strcompress(string(n_nodes),/remove_all)
& input_dat(3)='coordinates file name = '+name_nodes

 input_dat(4)='number         of         Dirichlet         nodes         =
'+strcompress(string(n_dirichlet),/remove_all)   &   input_dat(5)='Dirichlet
nodes file name = '+name_dirichlet

 input_dat(6)='number         of         Neumann         elements         =
'+strcompress(string(n_neumann),/remove_all)    &    input_dat(7)='Neumann
elements file name = '+name_neumann

 input_dat(8)='number       of       sources/sinks       elements       =
'+strcompress(string(n_sources_sinks),/remove_all)                        &
input_dat(9)='Sources/sinks elements file name = '+name_sources_sinks

 input_dat(10)='indexing             type             (0/1)             =
'+strcompress(string(index_type),/remove_all)

 input_dat(11)='solution    output    file    name    =    '+name_pot    &
input_dat(12)='velocity output file name = '+name_vel



status=dc_write_free(strcompress(PDE_Path+name_input,/remove_all),input_dat
,/column)

 cd,'c:\mmt\modeling\PDE'       &       spawn,'potential_flow.exe       '+
name_input,/console

 spawn,strcompress('copy     '+strcompress(PDE_path+name_pot,/remove_all)+'
'+DC_path)

 spawn,strcompress('copy     '+strcompress(PDE_path+name_vel,/remove_all)+'
'+DC_path)


end

;}}}

;{{{CreateShiftedArrays
```

```
Function mmts_CreateShiftedArrays,sarray,r,name,domains,acu

 dd=where(domains(0,*) EQ name) & ds=sort(domains(4,dd))
 if ds(0) EQ -1 then print,'domain problem in '+name

 for j=0,n_elements(ds)-1 do begin
  x=float(strsplit(domains(2,ds(j)),';'))                                    &
y=float(strsplit(domains(3,ds(j)),';'))
  pdata=mmts_Pdata(x,y) & pdata(2:3,*)=pdata(2:3,*)+0.25
  if j EQ 0 then dir=['S','E','N','W'] else dir=['N','W','S','E']
  for i=0,n_elements(pdata(0,*))-2 do begin
   x0=nint(pdata(2,i)*2.0)        &        y0=nint(pdata(3,i)*2.0)        &
x1=nint(pdata(2,i+1)*2.0) & y1=nint(pdata(3,i+1)*2.0)
   case 1 of
    x1 GT x0: sarray(x0:x1,y0:y0,5)= sarray(x0:x1,y0:y0,5)+dir(0) & y1 GT
y0: sarray(x0:x0,y0:y1,5)= sarray(x0:x0,y0:y1,5)+dir(1)
    x0 GT x1: sarray(x1:x0,y0:y0,5)= sarray(x1:x0,y0:y0,5)+dir(2) & y0 GT
y1: sarray(x0:x0,y1:y0,5)= sarray(x0:x0,y1:y0,5)+dir(3)
   else: print,'problem in domain section' & endcase
  endfor
 endfor

 acu_on=where(acu(13,*) EQ 'YES')
 if acu_on(0) GT -1 then begin
  for i=0,n_elements(acu_on)-1 do begin
   x0=nint(float(acu(1,acu_on(i))))+1                                        &
x1=x0+nint(float(acu(4,acu_on(i))))
   y0=nint(float(acu(2,acu_on(i))))+1                                        &
y1=y0+nint(float(acu(5,acu_on(i))))
   sarray(x0-1:x1,y0-1:y1,4)=acu(0,acu_on(i))
  endfor
 endif
 return,sarray
end
;}}}
;{{{Zoning
Function
mmts_Zoning,sarray,r,nsteps=nsteps,stepsize=stepsize,ShowTrajectory=ShowTra
jectory,freq=freq
 dxy=size(sarray) & x1=(dxy(1)-2)/2 & y1=(dxy(2)-2)/2
 if not keyword_set(nsteps) then nsteps=600
 if not keyword_set(stepsize) then stepsize=2.0/float(r)
 if not keyword_set(freq) then freq=5
```

```
 mask=shift(sarray(*,*,0),-1,-1) & mask=bytscl(float(mask))
failures=0 & fx=fltarr(8) & fy=fltarr(8) & xydata=fltarr(2,8) &
xt=fltarr(nsteps) & yt=fltarr(nsteps) & vec=sarray(*,*,0:3) &
vec=float(vec)
 zone=strarr(2*xl+1,2*yl+1)
 ix=0 & iy=0 & count=0
 cx=1 & cy=1
 repeat begin & ix=0 & repeat begin
  if mask(ix,iy) GT 0 then begin
   case 1 of
    mask(ix+1,iy+1) GT 0:
 begin & x=float(ix/2.0)+0.5 & y=float(iy/2.0)+0.5 & cx=2 & cy=2 & end
    mask(ix+1,iy) GT 0 and mask(ix+1,iy+1) EQ 0: begin & x=float(ix/2.0)
+0.5 & y=float(iy/2.0)+0.25 & cx=2 & cy=1 & end
    mask(ix,iy+1) GT 0 and mask(ix+1,iy+1) EQ 0: Begin &
x=float(ix/2.0)+0.25 & y=float(iy/2.0)+0.5 & cx=1 & cy=2 & end
   else: begin x=float(ix/2.0)+0.25 & y=float(iy/2.0)+0.25 & cx=1 & cy=1 &
end
   endcase

   xt(0)=x & yt(0)=y & steps=1 & acu_name='ZZ' & jumps=0

   repeat begin ;{{{
    vx0=nint(2.0*xt(steps-1)) & vy0=nint(2.0*yt(steps-1))
    case 1 of
     strpos(sarray(vx0,vy0,5),'N') GT -1: begin & vy0=vy0+1 &
yt(steps)=yt(steps-1)+0.5 & xt(steps)=xt(steps-1) & jumps=jumps+1& end
     strpos(sarray(vx0,vy0+1,5),'S') GT -1: begin & vy0=vy0-1 &
yt(steps)=yt(steps-1)-0.5 & xt(steps)=xt(steps-1) & jumps=jumps+1& end
     strpos(sarray(vx0,vy0,5),'E') GT -1: begin & vx0=vx0+1 &
xt(steps)=xt(steps-1)+0.5 & yt(steps)=yt(steps-1) & jumps=jumps+1& end
     strpos(sarray(vx0+1,vy0,5),'W') GT -1: begin & vx0=vx0-1 &
xt(steps)=xt(steps-1)-0.5 & yt(steps)=yt(steps-1) & jumps=jumps+1& end
    else: begin
     xydata(0,*)=[vx0-1.0+1.0/3.0,vx0-1.0+2.0/3.0,vx0-1.0+4.0/3.0,vx0-
1.0+5.0/3.0,vx0-1.0+1.0/3.0,vx0-1.0+2.0/3.0,vx0-1.0+4.0/3.0,vx0-
1.0+5.0/3.0]
     xydata(1,*)=[vy0-1.0+1.0/3.0,vy0-1.0+2.0/3.0,vy0-1.0+1.0/3.0,vy0-
1.0+2.0/3.0,vy0-1.0+4.0/3.0,vy0-1.0+5.0/3.0,vy0-1.0+4.0/3.0,vy0-
1.0+5.0/3.0]
     fx(0)=vec(vx0,vy0,0) & fx(1)=vec(vx0,vy0,1) & fx(2)=vec(vx0+1,vy0,0)
& fx(3)=vec(vx0+1,vy0,1)
     fx(4)=vec(vx0,vy0+1,0) & fx(5)=vec(vx0,vy0+1,1) &
fx(6)=vec(vx0+1,vy0+1,0) & fx(7)=vec(vx0+1,vy0+1,1)
```

```
    fy(0)=vec(vx0,vy0,2) & fy(1)=vec(vx0,vy0,3)  & fy(2)=vec(vx0+1,vy0,2)
& fy(3)=vec(vx0+1,vy0,3)
    fy(4)=vec(vx0,vy0+1,2)      &      fy(5)=vec(vx0,vy0+1,3)            &
fy(6)=vec(vx0+1,vy0+1,2) & fy(7)=vec(vx0+1,vy0+1,3)
    vx=avg(fx) & vy=avg(fy)
    ;vx=SCAT2DINTERP(xydata,  fx,  [2.0*xt(steps-1)],[2.0*yt(steps-1)])  &
vy=SCAT2DINTERP(xydata, fy, [2.0*xt(steps-1)],[2.0*yt(steps-1)])
    v=sqrt(vx^2+vy^2) & n=0.2/v & xt(steps)=(xt(steps-1)+vx*stepsize*n) &
yt(steps)=(yt(steps-1)+vy*stepsize*n)
   endelse & endcase
   if sarray(vx0,vy0,4) NE '' then acu_name=sarray(vx0,vy0,4)
     ;if zone(vx0,vy0) NE '' then acu_name=zone(vx0,vy0)
   steps=steps+1 ;;;}}}
  endrep until steps EQ nsteps or acu_name NE 'ZZ'

  if acu_name EQ 'ZZ' then failures=failures+1
  zone(ix+1:ix+cx,iy+1:iy+cy)=acu_name
  if  keyword_set(ShowTrajectory)  and  count  EQ  freq  then  begin  &
oplot,xt(0:steps-1),yt(0:steps-1),color=4,thick=0.5,Nsum=5  &  count=0  &
endif
  count=count+1
 endif
ix=ix+cx & endrep until ix GE 2*xl-1 & iy=iy+cy & endrep until iy GT 2*yl-
1
zone=shift(zone,-1,-1) & zone=zone(0:2*xl-1,0:2*yl-1)
return,zone
end
;}}}
;{{{CreateZonePolyLines
Function mmts_CreateZonePolyLines,pzone,crac
dxy=size(pzone) & xl=dxy(1)/2 & yl=dxy(2)/2
intzone1=intarr(2*xl,2*yl)      &      intzone=intarr(2*xl+2,2*yl+2)      &
outz=strarr(500) & q=0
for i=1,n_elements(crac)-1 do begin
 intzone1(*,*)=0  &  intzone(*,*)=0  &  intzone1(where(pzone  EQ  i))=i  &
intzone(0:2*xl-1,0:2*yl-1)=intzone1
 intzone=intzone+shift(intzone,1,0)+shift(intzone,0,1)+shift(intzone,1,1)
& intzone(where(intzone GE i))=i
 p=boundary(intzone,where(intzone       EQ       i),k=2)       &       x=p-
great_int(p/(2*xl+2))*(2*xl+2) & y=great_int(p/(2*xl+2))
 data=intarr(2,n_elements(x)+50)    &    data(0,0:n_elements(x)-1)=x    &
data(1,0:n_elements(x)-1)=y
 n=0
                          ; find center points
```

```
  for c=0,n_elements(x)-1 do begin
   axf=where(data(0,c)+1 EQ data(0,*) and data(1,c) EQ data(1,*)) &
axb=where(data(0,c)-1 EQ data(0,*) and data(1,c) EQ data(1,*))
   ayf=where(data(0,c) EQ data(0,*) and data(1,c)+1 EQ data(1,*)) &
ayb=where(data(0,c) EQ data(0,*) and data(1,c)-1 EQ data(1,*))
   if axf(0) GT -1 and axb(0) GT -1 and ayf(0) GT -1 and ayb(0) GT -1 then
begin data(*,n_elements(x)+n)=data(*,c) & n=n+1 & endif
  endfor
  data=data(*,0:n_elements(x)+n) & x=data(0,*) & y=data(1,*) & c=0

  repeat begin
   seed=c                                                                &
ind=mmts_FindConnectingVertices(seed,data=data,c=c,status=status)
   t=strcompress(string(x(ind))+','+string(y(ind)),/remove_all)
   out='<cracz id="'+crac(i)+'" bounds="'+strjoin(t,',')+',"/>'
   if n_elements(t) GT 2 then begin
   outz(q)=out & q=q+1 & endif
  endrep until status EQ 'finished'
 endfor
 outz=outz(0:q-1)
 return,outz
end
;}}}

 if keyword_set(cool) then begin
  mmts_OpenDevice2D,strcompress(DC_path+'\cool.ps'),xl,yl
  mmts_Page2D,xl,yl,banner=banner,cmap=cmap,title=strcompress('Cooling
Capacity - '+general(2))
  dis=gcool
  mmts_DisplayGArray2D,dis,xl,yl,high=max(gcool),low=min(gcool)

mmts_DrawLayout2D,['PLENUM','PERF','WALL','STRUCTURE','ACU','SERVER','FURNI
TURE','TAPE_OPTICAL','DISK','PDU','NETWORK','TILES'],glayout,xl,yl
  mmts_LabelEquipment2D,['PDU'],glayout
  mmts_DrawInlet2D,['SERVER','DISK','NETWORK'],glayout

mmts_LabelVane2D,float(acu(1,*)),float(acu(2,*)),float(acu(4,*)),float(acu(
5,*)),acu(14,*)
  mmts_DisplayColorbar,xl/2+1+1,3,ct,max(gcool),min(gcool),'Cooling   Power
[kW]'

  mmts_Page3D,xl,yl,zl,banner=banner,cmap=cmap,title='Cooling Capacity (3D)
- '+general(2),wn=2
```

```
mmts_DrawLayout2D,['PLENUM','PERF','WALL','STRUCTURE','ACU','SERVER','FURNI
TURE','TAPE_OPTICAL','DISK','PDU','NETWORK','TILES'],glayout,xl,yl
  mmts_DrawInlet2D,inlets,glayout
  mmts_LabelEquipment2D,labels,glayout
  LOADCT,22,/Silent & tek_color & shades=gcool ;& shades(where(shades LE
100))=100 ;& shades=bytscl((shades-low_flow)/(high_flow-low_flow) )
  shades=bytscl(shades)

mmts_bar3d,gcool,/noerase,ZRange=[0,max(gcool)*8],threshold=min(gcool)+0.01
,XRange=[-8,xl],YRange=[-
8,yl],IndividualColors=bytscl(shades),/Outline,color=1,/noshade
  mmts_DisplayColorbar,xl/2+1+1,15,ct,max(gcool),min(gcool),'Cool
[kW]',dx=1.0
  device,/close
 endif
;}}}
;*** FE GENERAL SETTINGS {{{
 print,'*** PLENUM FE MODELING ***'
 name='PD1'    &    r=2.0    &         dom_nodes=lonarr(r*xl+1,r*yl+1)    &
dom_triangles=strarr(nint(r*xl),nint(r*yl))
; obstruct=['STRUCTURE']
;}}}
;*** FE MESH SECTION 2D PLENUM {{{
 if keyword_set(mesh) then begin
  mmts_OpenDevice2D,strcompress(DC_path+'\mesh.ps',/Remove_all),xl,yl
  mmts_Page2D,xl,yl,banner=banner,cmap=cmap,title='Layout                -
'+general(2),wn=0
  mmts_DrawLayout2D,['WALL','TILES'],glayout,xl,yl

mmts_FeAutoMesherNodes2D,name,domains,r,glayout,obstruct=obstruct,dom_nodes
=dom_nodes,nodes=nodes,total_area=area,/Show_nodes,/Show_Boundary

mmts_FeAutoMesherTriangles2D,r,nodes,dom_nodes,dom_triangles=dom_triangles,
triangles=triangles,/Show_triangles

mmts_FeSourcesSinks,acu_b,acuMetrics_b,gflow,dom_triangles,r,sources_sinks=
sources_sinks,/Show_sources,/Show_sinks
  xyouts,xl+0.05,-1.3,strcompress('Area:              '+string(nint(area))+
'ft2'),color=0,Size=1.2
  device,/close
 endif else begin

mmts_FeAutoMesherNodes2D,name,domains,r,glayout,obstruct=obstruct,dom_nodes
=dom_nodes,nodes=nodes,total_area=area
```

```
mmts_FeAutoMesherTriangles2D,r,nodes,dom_nodes,dom_triangles=dom_triangles,
triangles=triangles

mmts_FeSourcesSinks,acu_b,acuMetrics_b,gflow,dom_triangles,r,sources_sinks=
sources_sinks
 endelse
;}}}
;*** INVOKE SOLVER & GET VEC {{{
 ai=where(domains(0,*) EQ name and domains(4,*) EQ 'inner')
 if ai(0) EQ -1 or n_elements(ai) NE 1 then print,'Domain error'
 if checkfile(DC_path+'\~pot.txt',/read) EQ 0 then begin
  dirichlet=domains(5,ai(0))                                          &
strput,dirichlet,',',strpos(domains(5,0),';')
  neumann=0 & index_type= 1
  PDE_path='c:\mmt\modeling\PDE\ '

mmts_FESolver,PDE_Path,DC_path,triangles,nodes,dirichlet,neumann,sources_si
nks,index_type
 endif
 status=dc_read_free(strcompress(DC_path+'\~vel.txt',/remove_all),vecxyz)  &
vecxyz=reform(vecxyz,4,n_elements(vecxyz)/4)
 vecxyz(2,*)=vecxyz(2,*)*(-1) & vecxyz(3,*)=vecxyz(3,*)*(-1)
;}}}
;*** POT + VEC SECTION {{{
 if keyword_set(potvec) then begin
  pot=fltarr(r*xl+1,r*yl+1) & pot(*,*)=0
  status=dc_read_free(strcompress(DC_path+'\~pot.txt',/remove_all),potxyz)
& potxyz=reform(potxyz,3,n_elements(potxyz)/3)
  pot(nint(potxyz(0,*)),nint(potxyz(1,*)))=potxyz(2,*)
  ai=where(pot   EQ   0)   &   dis=pot   &   dis(ai(*))=min(pot)-(max(pot)-
min(pot))/250.0
  mmts_OpenDevice2D,strcompress(DC_path+'\potvec.ps',/Remove_all),xl,yl
  mmts_Page2D,xl,yl,banner=banner,cmap=cmap,title='Flow  Potential  +  Vec
field - '+general(2),wn=0
  mmts_DisplayGArray2D,dis,xl,yl,ct=ct,r=r,/nodes
  mmts_DrawLayout2D,['WALL','TILES','PERF'],glayout,xl,yl
  xyouts,xl+0.05,-1.3,strcompress('Area:              '+string(nint(area))+
'ft2'),color=0,Size=1.2
  ;plots,pdata(2:3,*),color=3,thick=3,LineStyle=2
  for i=0,n_elements(acu_b(0,*))-1 do begin
   ai=i   &   x=float(acu_b(1,ai))/2.0   &   y=float(acu_b(2,ai))/2.0   &
dx=float(acu_b(4,ai))/2.0 & dy=float(acu_b(5,ai))/2.0
   plots,[x,x+dx,x+dx,x,x],[y,y,y+dy,y+dy,y],color=4,thick=5 & endfor
  ;oplot,[vecxyz(0,*)/2.0],[vecxyz(1,*)/2.0],psym=4,color=2,symsize=0.25
```

```
  for                i=long(0),n_elements(vecxyz(0,*))-1,2                do
mmts_Vector2D,vecxyz(0,i)/2.0,vecxyz(1,i)/2.0,-vecxyz(2,i),-
vecxyz(3,i),0,1.0,length=0.7,/center
  device,/close
 endif
;}}}
;*** PLEN + VEC SECTION {{{
 if keyword_set(plenvec) then begin
  mmts_OpenDevice2D,strcompress(DC_path+'\plenvec.ps',/Remove_all),xl,yl
  mmts_Page2D,xl,yl,banner=banner,cmap=cmap,title='Plenum Temp + Vec field
- '+general(2),wn=0
  dis=gplenum
  mmts_DisplayGArray2D,dis,xl,yl,ct=ct
  for                i=long(0),n_elements(vecxyz(0,*))-1,2                do
mmts_Vector2D,vecxyz(0,i)/2.0,vecxyz(1,i)/2.0,-vecxyz(2,i),-
vecxyz(3,i),0,1.0,length=0.7,/center

mmts_DrawLayout2D,['WALL','PERF','SERVER','DISK','NETWORK','ACU','PDU','TAP
E_OPTICAL','TILES'],glayout,xl,yl
  mmts_DrawInlet2D,['SERVER','DISK','NETWORK'],glayout
  xyouts,xl+0.05,-1.3,strcompress('Area:              '+string(nint(area))+
'ft2'),color=0,Size=1.2
  ;plots,pdata(2:3,*),color=3,thick=3,LineStyle=2

mmts_DisplayColorbar,xl/2+1+1,3,ct,max(gplenum)*1.8+32.0,min(gplenum)*1.8+3
2.0,'Temp [F]'
  ;oplot,[vecxyz(0,*)/2.0],[vecxyz(1,*)/2.0],psym=4,color=2,symsize=0.25
  device,/close
 endif
;}}}
;*** INVOKE ZONING {{{
 print,'*** ZONING ***'
 sarray=strarr(r*xl+2,r*yl+2,6)     ;vecx - first cell / second cell

vecy - first cell / second cell
 sarray(*,*,0:3)='-999'
 sarray(great_int(vecxyz(0,*))+1,great_int(vecxyz(1,*))+1,nint(vecxyz(0,*)-
great_int(vecxyz(0,*)))) = vecxyz(2,*)

sarray(great_int(vecxyz(0,*))+1,great_int(vecxyz(1,*))+1,2+nint(vecxyz(0,*)
-great_int(vecxyz(0,*)))) = vecxyz(3,*)
 if checkfile(DC_path+'\~zone.txt',/read) EQ 0 then begin
  sarray=mmts_CreateShiftedArrays(sarray,r,name,domains,acu_b)
  nsteps=600 & stepsize=2.0/float(r)
  zone=mmts_Zoning(sarray,r,nsteps=nsteps,stepsize=stepsize)
```

```
status=dc_write_free(strcompress(DC_path+'\~zone.txt',/remove_all),zone,/co
lumn)
 endif else begin
 zone=strarr(2*xl,2*yl)                                                    &
status=dc_read_free(strcompress(DC_path+'\~zone.txt',/remove_all),zone,/col
umn)
 endelse
 ai=where(zone EQ 'ZZ') & if ai(0) NE -1 then zone(ai)=''
 crac=unique(zone(*,*)) & crac=crac(sort(crac)) & pzone=intarr(2*xl,2*yl)
 for i=0,n_elements(crac)-1 do pzone(where(zone EQ crac(i)))=i
 pzone=fix(pzone)
 ; }}}
;*** CREATE ZONE POLYLINES {{{
 outz=mmts_CreateZonePolyLines(pzone,crac)
 zones_out=strarr(n_elements(outz))
 if keyword_set(cracz) then begin

status=dc_write_fixed(strcompress(DC_path+'\~cracz.txt',/remove_all),outz)
  t=strarr(n_elements(outz))    &    crac_id=strarr(n_elements(outz))    &
pss=strarr(n_elements(outz))
  for i=0,n_elements(outz)-1 do begin
   s=strpos(outz(i),'bounds')+8      &      e=strpos(outz(i),'"/>')      &
t(i)=strcompress(strmid(outz(i),s,e-s-1),/remove_all)
   s=strpos(outz(i),'<cracz  id="')+11  &  e=strpos(outz(i),'"  bounds')  &
crac_id(i)=strcompress(strmid(outz(i),s,e-s),/remove_all)
   zones_out(i)=strcompress(crac_id(i)+','+t(i)+',',/remove_all)
  endfor
  ;here write polylines to dB
  class='CRAC_ZONES' & columns='id,bounds'
  args = 'FRMT=CSV&LST=' + class + ',' + columns
  cache = '\layout&' + args
  file = FILEPATH(args, /Tmp) + '.csv'
  status = dc_write_fixed(file,zones_out)
  resp = mmtsdb(sessionContext, '/api/layout/properties/set', Args=args,
Cache=cache, UploadFile=file)
 endif
;}}}
;*** ZONE SECTION {{{
 if keyword_set(zones) then begin
  mmts_OpenDevice2D,strcompress(DC_path+'\zone.ps'),xl,yl
  mmts_Page2D,xl,yl,banner=banner,cmap=cmap,title=strcompress('ACU    Zones
'+general(2))
  mmts_DisplayGArray2D,pzone,xl,yl,high=max(pzone),low=min(pzone)
```

```
  mmts_DrawLayout2D,['TILES','PERF','ACU','WALL'],glayout,xl,yl
  for                i=long(0),n_elements(vecxyz(0,*))-1,4                do
mmts_Vector2D,vecxyz(0,i)/2.0,vecxyz(1,i)/2.0,vecxyz(2,i),vecxyz(3,i),0,1.0
,length=0.7,/center
  for i=0,n_elements(acu_off_m)-1 do begin
   x=float(acu(1,acu_off_m(i)))/2.0  &  y=float(acu(2,acu_off_m(i)))/2.0  &
dx=float(acu(4,acu_off_m(i)))/2.0 & dy=float(acu(5,acu_off_m(i)))/2.0
   if  dx GE dy then ori=0 else ori=90

polyfill,[x,x+dx,x+dx,x,x],[y,y,y+dy,y+dy,y],color=2,/Line_fill,orientation
=45,spacing=0.2,thick=1.0
   plots,[x,x+dx,x+dx,x,x],[y,y,y+dy,y+dy,y],color=2,thick=2
  endfor

  for i=0,n_elements(outz)-1 do begin
   s=strpos(outz(i),'bounds')+8 & e=strpos(outz(i),'"/>')
   t=strsplit(strmid(outz(i),s,e-s),',') & ind=indgen(n_elements(t)/2)*2
   plots,float(t(ind))/2.0,float(t(ind+1))/2.0,thick=5,color=0
  endfor

 device,/close
 endif
 ;}}}
;*** TRAJECTORY SECTION {{{
 if keyword_set(trajectory) then begin
  mmts_OpenDevice2D,strcompress(DC_path+'\trajectory.ps'),xl,yl
  mmts_Page2D,xl,yl,banner=banner,cmap=cmap,title=strcompress('ACU     Zones
'+general(2))
  mmts_DisplayGArray2D,pzone,xl,yl,high=max(pzone),low=min(pzone)
  mmts_DrawLayout2D,['TILES','PERF','ACU','WALL'],glayout,xl,yl
  for                i=long(0),n_elements(vecxyz(0,*))-1,4                do
mmts_Vector2D,vecxyz(0,i)/2.0,vecxyz(1,i)/2.0,-vecxyz(2,i),-
vecxyz(3,i),0,1.0,length=0.7,/center

  for i=0,n_elements(acu_off_m)-1 do begin
   x=float(acu(1,acu_off_m(i)))/2.0  &  y=float(acu(2,acu_off_m(i)))/2.0  &
dx=float(acu(4,acu_off_m(i)))/2.0 & dy=float(acu(5,acu_off_m(i)))/2.0
   if  dx GE dy then ori=0 else ori=90

polyfill,[x,x+dx,x+dx,x,x],[y,y,y+dy,y+dy,y],color=2,/Line_fill,orientation
=45,spacing=0.2,thick=1.0
   plots,[x,x+dx,x+dx,x,x],[y,y,y+dy,y+dy,y],color=2,thick=2
  endfor
```

```
  nsteps=600 & stepsize=2.0/float(r)
  sarray=mmts_CreateShiftedArrays(sarray,r,name,domains,acu_b)
  freq=5                                                                    &
zone=mmts_Zoning(sarray,r,nsteps=nsteps,stepsize=stepsize,/ShowTrajectory,f
req=freq)

  for i=0,n_elements(outz)-1 do begin
   s=strpos(outz(i),'bounds')+8 & e=strpos(outz(i),'"/>')
   t=strsplit(strmid(outz(i),s,e-s),',') & ind=indgen(n_elements(t)/2)*2
   plots,float(t(ind))/2.0,float(t(ind+1))/2.0,thick=5,color=0
  endfor

;  tek_color
;  for x=0,2*xl+2-1 do begin &  for y=0,2*yl+2-1 do begin
;       xt=(x-1)/2.0  &  yt=(y-1)/2.0  &  if  sarray(x,y,5)  NE  ''  then
xyouts,xt,yt,strcompress(sarray(x,y,5)),charsize=0.5,color=0
;  endfor & endfor

  device,/close
 endif
;}}}
```

[0067] Although illustrative embodiments of the present invention have been described herein, it is to be understood that the invention is not limited to those precise embodiments, and that various other changes and modifications may be made by one skilled in the art without departing from the scope of the invention.

Claims

What is claimed is:

1. A method for modeling thermal zones in a space, comprising the steps of:

providing a graphical representation of the space;

defining at least one domain in the space for modeling;

creating a mesh in the domain by sub-dividing the domain into a set of discrete sub-domains that interconnect a plurality of nodes;

identifying air flow sources and sinks in the domain;

obtaining air flow measurements from one or more of the air flow sources and sinks;

determining an air flow velocity vector at a center of each sub-domain using the air flow measurement obtained from the air flow sources and sinks; and

tracing each velocity vector to one of the air flow sources, wherein a combination of the traces to a given one of the air flow sources represents a thermal zone in the space.

2. The method of claim 1, wherein the space comprises a room.

3. The method of claim 2, wherein the room comprises a data center having computer equipment racks and a raised-floor cooling system with one or more computer air conditioning units configured to take in hot air from the computer equipment racks and to exhaust cooled air into a sub-floor plenum that is delivered to the computer equipment racks through a plurality of perforated tiles in the raised floor.

4. The method of claim 3, wherein the domain is defined by the dimensions of the sub-floor plenum.

5. The method of claim 1, wherein the air flow measurements from the air flow sources and sinks are obtained using mobile measurement technology.

6. The method of claim 4, wherein the perforated tiles comprise air flow sinks and the air conditioning units comprise air flow sources.

7. The method of claim 6, further comprising the step of:

tracing each of the velocity vectors to one of the air conditioning units, wherein a combination of the traces to a given one of the air conditioning units represents a thermal zone in the space.

8. The method of claim 1, further comprising the steps of:

repeating the identifying, obtaining, using and tracing steps in response to a change in air flow at one or more of the air flow sources and sinks in the domain.

9. The method of claim 1, further comprising the step of:

repeating the identifying, obtaining, determining and tracing steps in response to one or more of the air flow sources and sinks being removed from the domain.

10. The method of claim 1, further comprising the step of:

repeating the identifying, obtaining, determining and tracing steps in response to one or more new air flow sources and sinks being added to the domain.

11. The method of claim 1, further comprising the step of:

determining a cooling capacity for each air flow source.

12. The method of claim 1, further comprising the step of:

periodically repeating the obtaining, determining and tracing steps to update the air flow measurements from the air flow sources and sinks.

13. The method of claim 1, wherein the tracing step further comprises the steps of:

selecting a given location in the space;

referencing the air flow velocity vector determined at the center of the sub-domain at the location;

determining a next location in the space;

repeating the steps of referencing the air flow velocity vector and determining a next location for n number of locations throughout the space, each time assigning the given location to a given air flow source when the next location falls within an area of that given air flow source.

14. The method of claim 13, further comprising the step of:

placing a limit on the number of times the repeating step is performed without having a location fall within the area of a given air flow source.

15. The method of claim 14, further comprising the step of:

determining whether the repeating step has been performed too many times, whenever the next location does not fall within the area of an air flow source.

16. The method of claim 14, further comprising the step of:

designating the given location as not being associated with a thermal zone whenever the limit is reached.

17. An article of manufacture for modeling thermal zones in a space, comprising a machine-readable medium containing one or more programs which when executed implement the steps of the method according to claim 1.

18. An apparatus for modeling thermal zones in a space, the apparatus comprising:

a memory; and

at least one processor device, coupled to the memory, operative to:

provide a graphical representation of the space;

define at least one domain in the space for modeling;

create a mesh in the domain by sub-dividing the domain into a set of discrete sub-domains that interconnect a plurality of nodes;

identify air flow sources and sinks in the domain;

obtain air flow measurements from one or more of the air flow sources and sinks;

determine an air flow velocity vector at a center of each sub-domain using the air flow measurement obtained from the air flow sources and sinks; and

trace each velocity vector to one of the air flow sources, wherein a combination of the traces to a given one of the air flow sources represents a thermal zone in the space.

FIG. 1

100

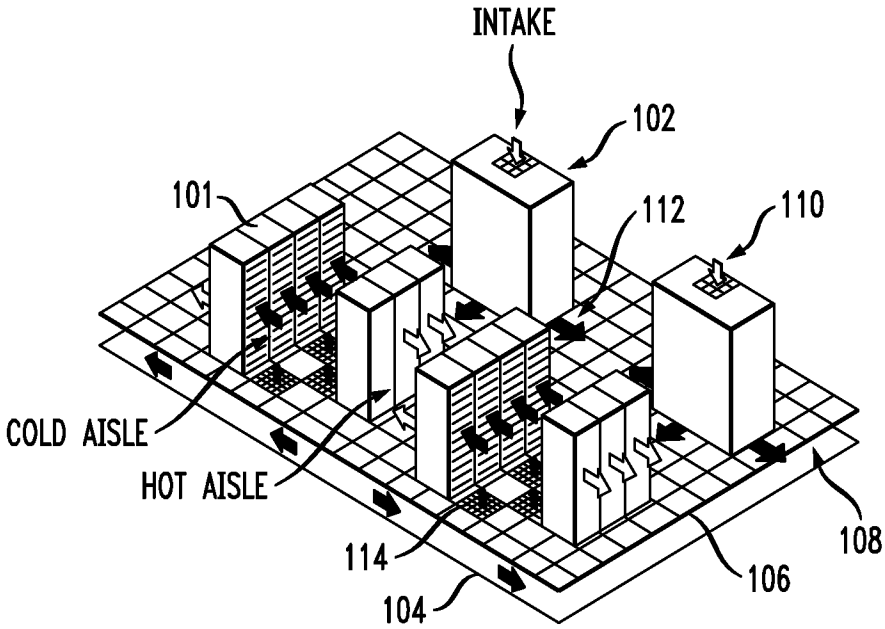


FIG. 2

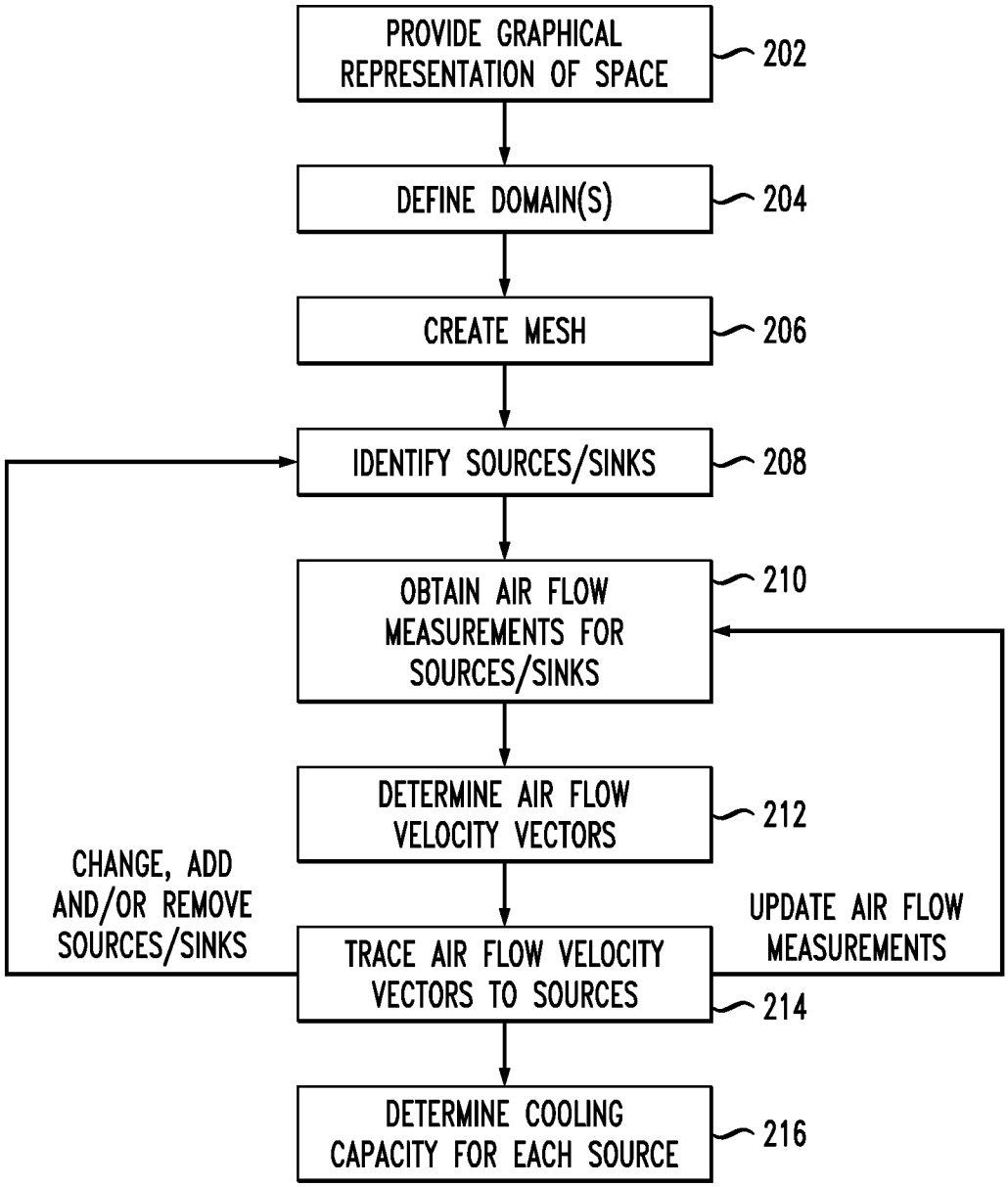
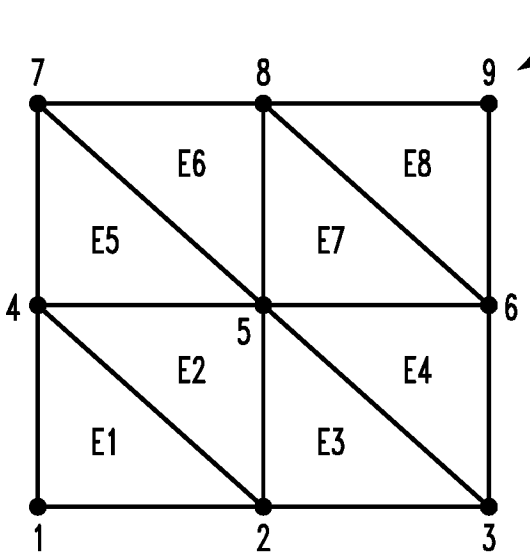

200
PROVIDE GRAPHICAL REPRESENTATION OF SPACE
202
DEFINE DOMAIN(S)
204
CREATE MESH
206
IDENTIFY SOURCES/SINKS
208
OBTAIN AIR FLOW MEASUREMENTS FOR SOURCES/SINKS
210
DETERMINE AIR FLOW VELOCITY VECTORS
212
CHANGE, ADD AND/OR REMOVE SOURCES/SINKS
UPDATE AIR FLOW MEASUREMENTS
TRACE AIR FLOW VELOCITY VECTORS TO SOURCES
214
DETERMINE COOLING CAPACITY FOR EACH SOURCE
216

FIG. 3



A MESH CONSISTING OF A SET OF NINE NODES AND EIGHT ELEMENTS

● : INDICATES A NODE

ELEMENTS	CORRESPONDING NODES
E1	1, 2, 4
E2	4, 2, 5
E3	2, 3, 5
E4	3, 6, 5
E5	4, 5, 7
E6	5, 8, 7
E7	5, 6, 8
E8	6, 9, 8

FIG. 4

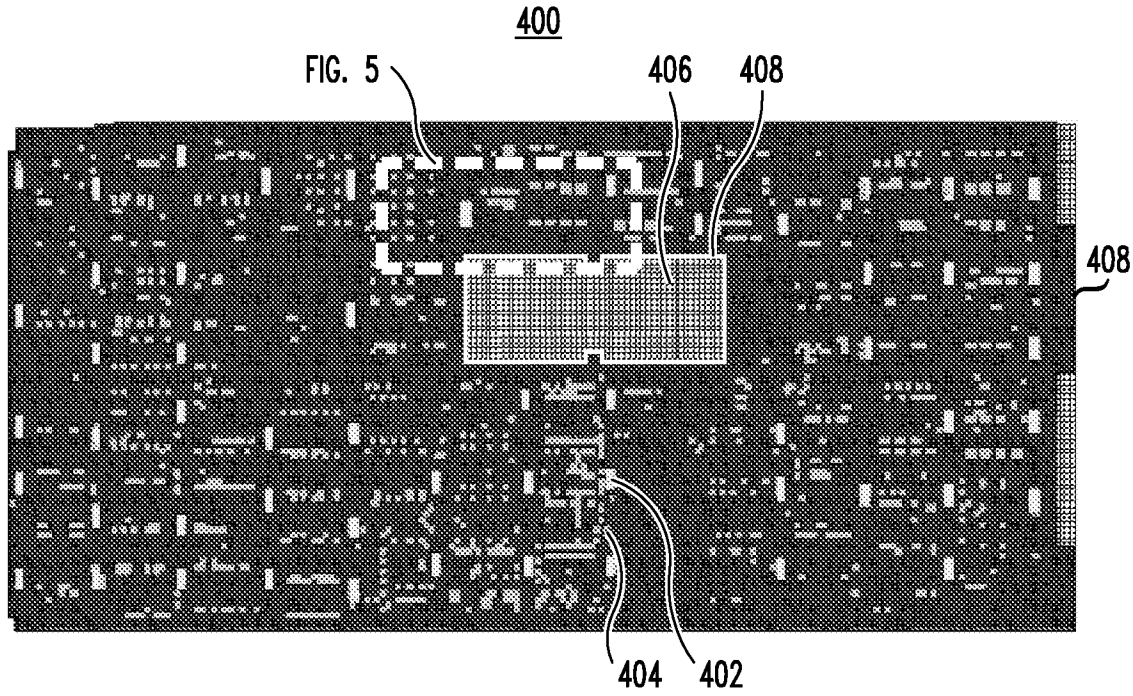


FIG. 5

500

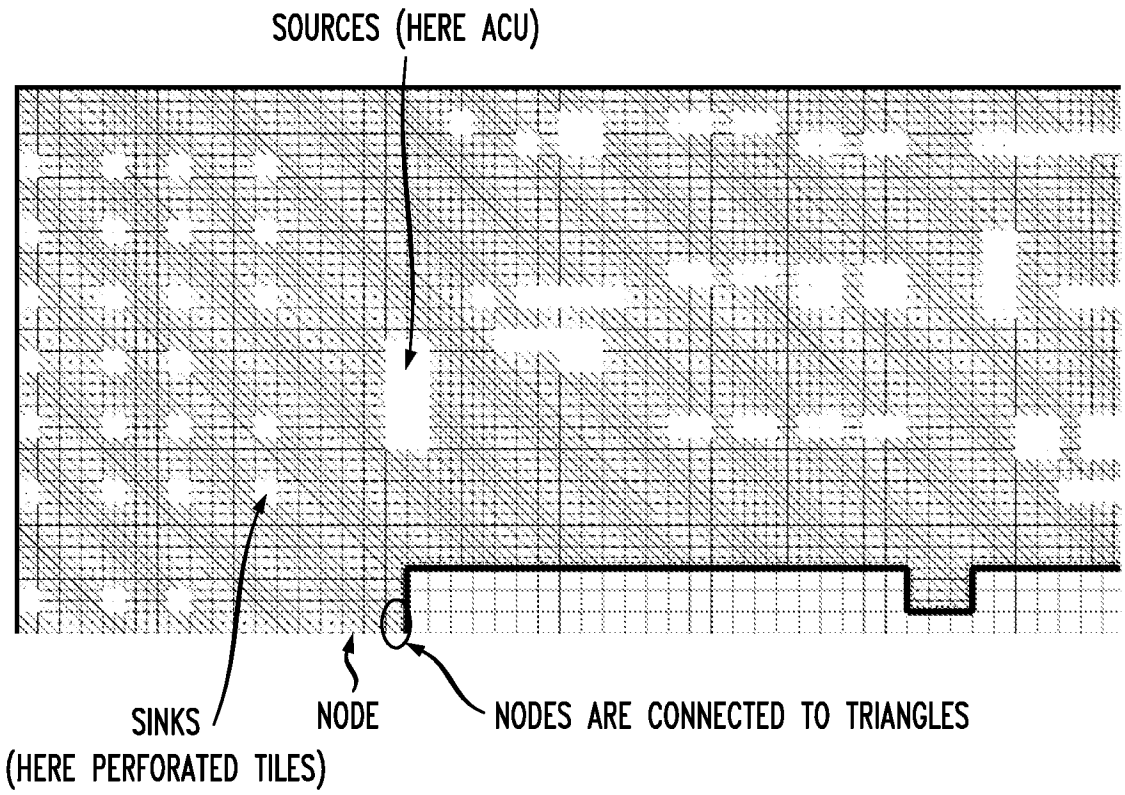
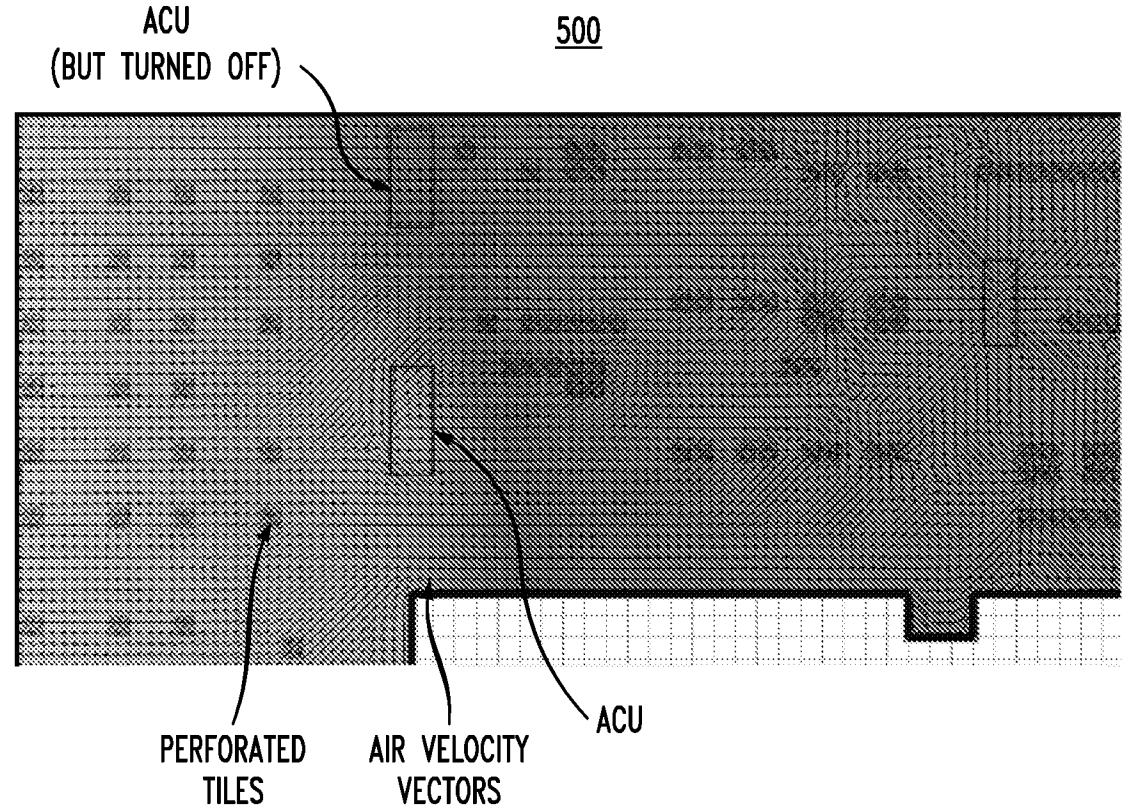


FIG. 6

500



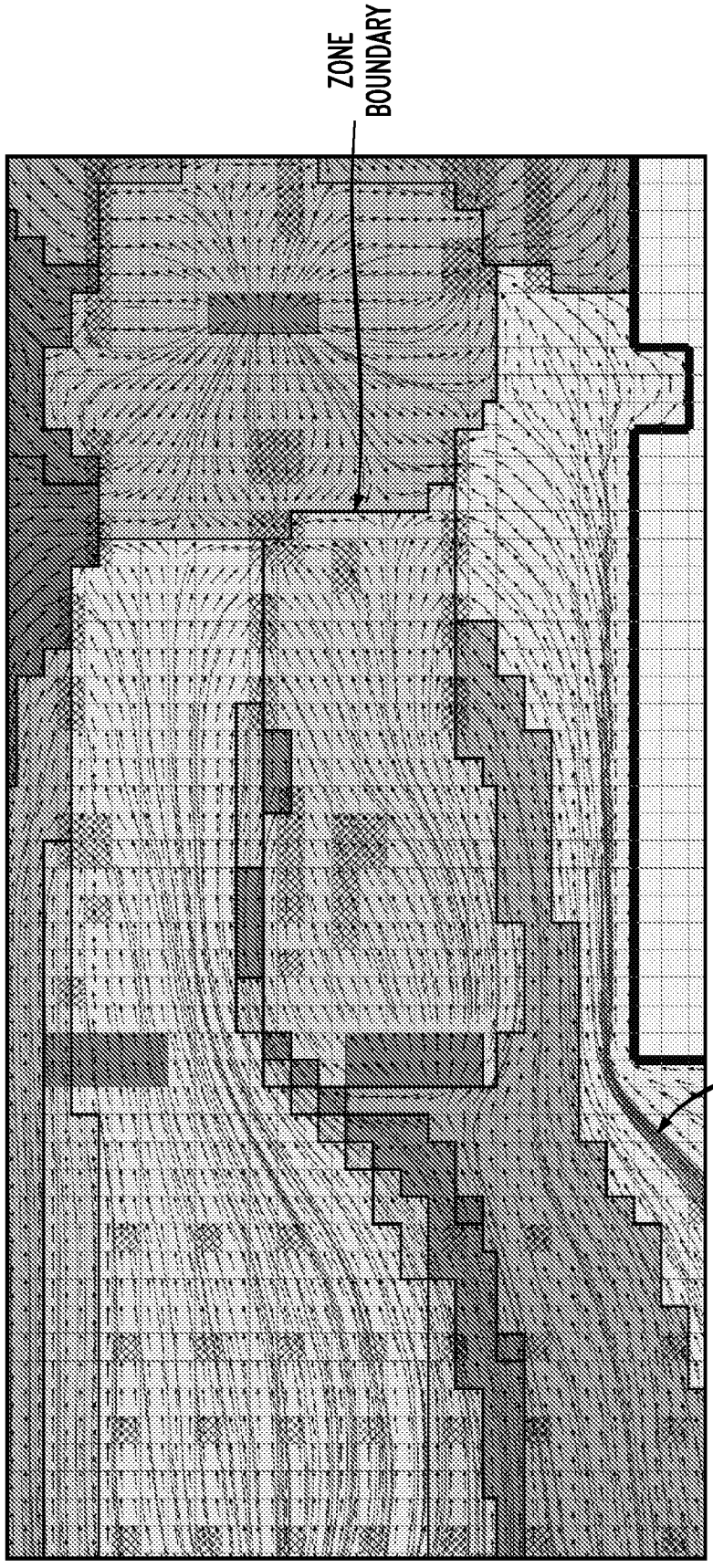


FIG. 7

500

FIG. 8

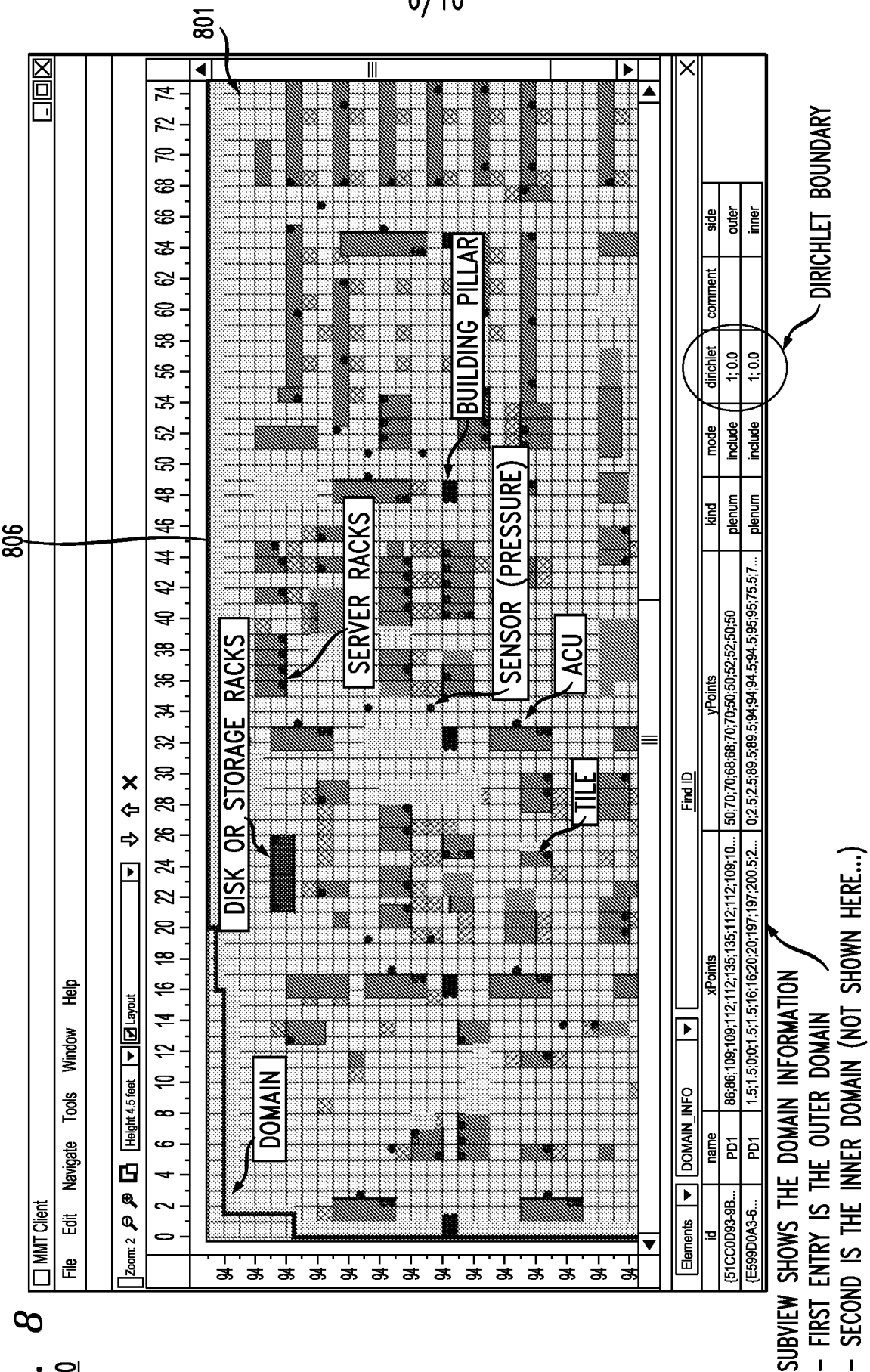
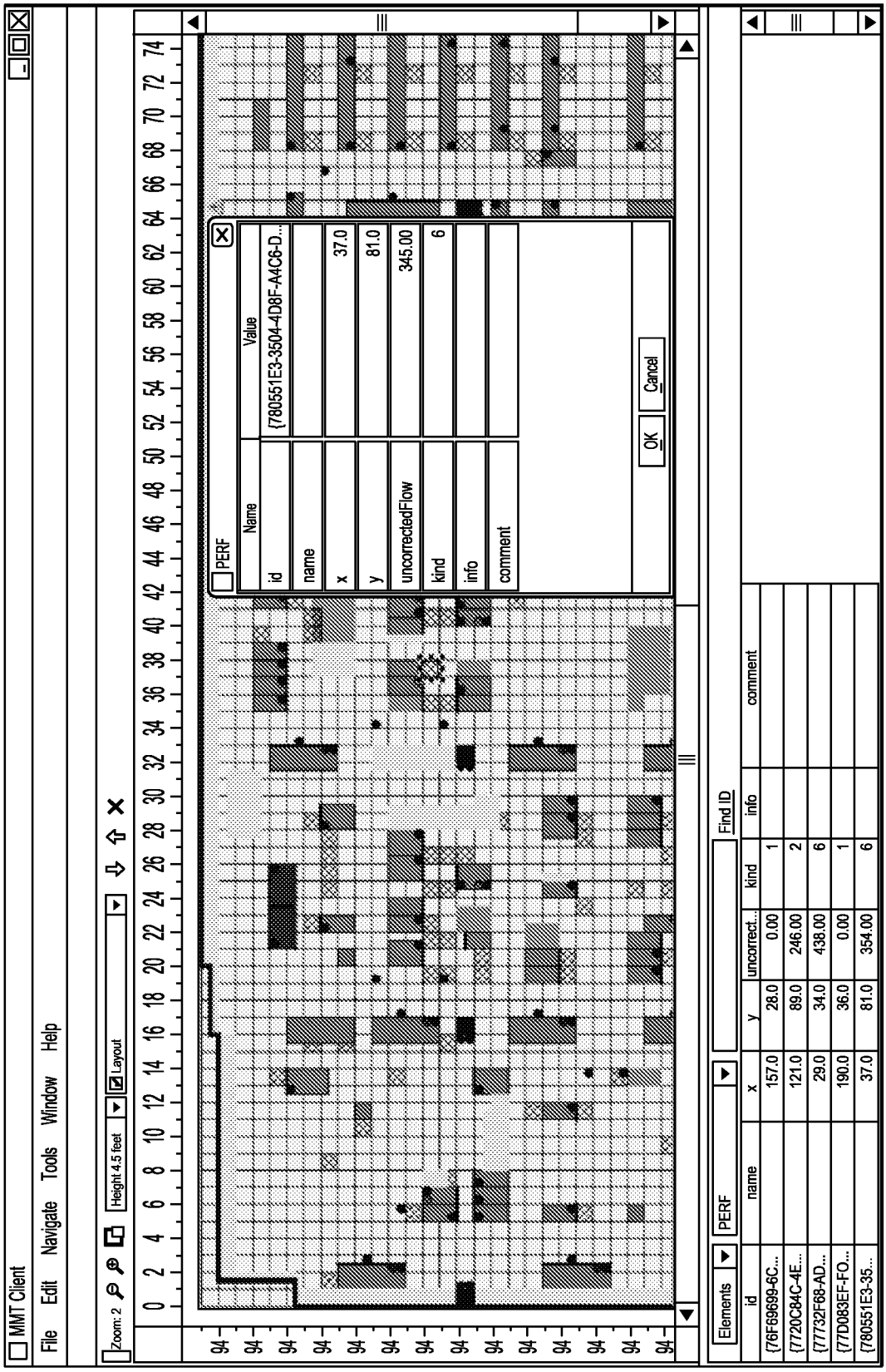

800
801
806
MMT Client
File Edit Navigate Tools Window Help
Zoom: 2
Height 4.5 feet
Layout
DISK OR STORAGE RACKS
SERVER RACKS
BUILDING PILLAR
SENSOR (PRESSURE)
ACU
TILE
DOMAIN
Elements
DOMAIN_INFO
Find ID
id name xPoints yPoints kind mode dirichlet comment side
{51CC0D93-9B... PD1 86;86;109;109;112;112;135;135;112;112;109;10... 50;70;70;68;68;70;70;50;50;52;52;50;50 plenum include 1; 0.0 outer
{E599D0A3-6... PD1 1.5;1.5;0;0;1.5;1.5;16;16;20;20;197;197;200.5;2... 0;2.5;2.5;89.5;89.5;94;94;94.5;94.5;95;95;75.5;7... plenum include 1; 0.0 inner
DIRICHLET BOUNDARY
SUBVIEW SHOWS THE DOMAIN INFORMATION
- FIRST ENTRY IS THE OUTER DOMAIN
- SECOND IS THE INNER DOMAIN (NOT SHOWN HERE...)

FIG. 9


800
MMT Client
File
Edit
Navigate
Tools
Window
Help
Zoom: 2
Height 4.5 feet
Layout
PERF
Name
Value
id
{780551E3-3504-4D8F-A4C6-D...
name
x
37.0
y
81.0
uncorrectedFlow
345.00
kind
6
info
comment
OK
Cancel
Elements
PERF
Find ID
id
name
x
y
uncorrect...
kind
info
comment
{76F69699-6C...
157.0
28.0
0.00
1
{7720C84C-4E...
121.0
89.0
246.00
2
{77732F68-AD...
29.0
34.0
438.00
6
{77D083EF-FO...
190.0
36.0
0.00
1
{780551E3-35...
37.0
81.0
354.00
6

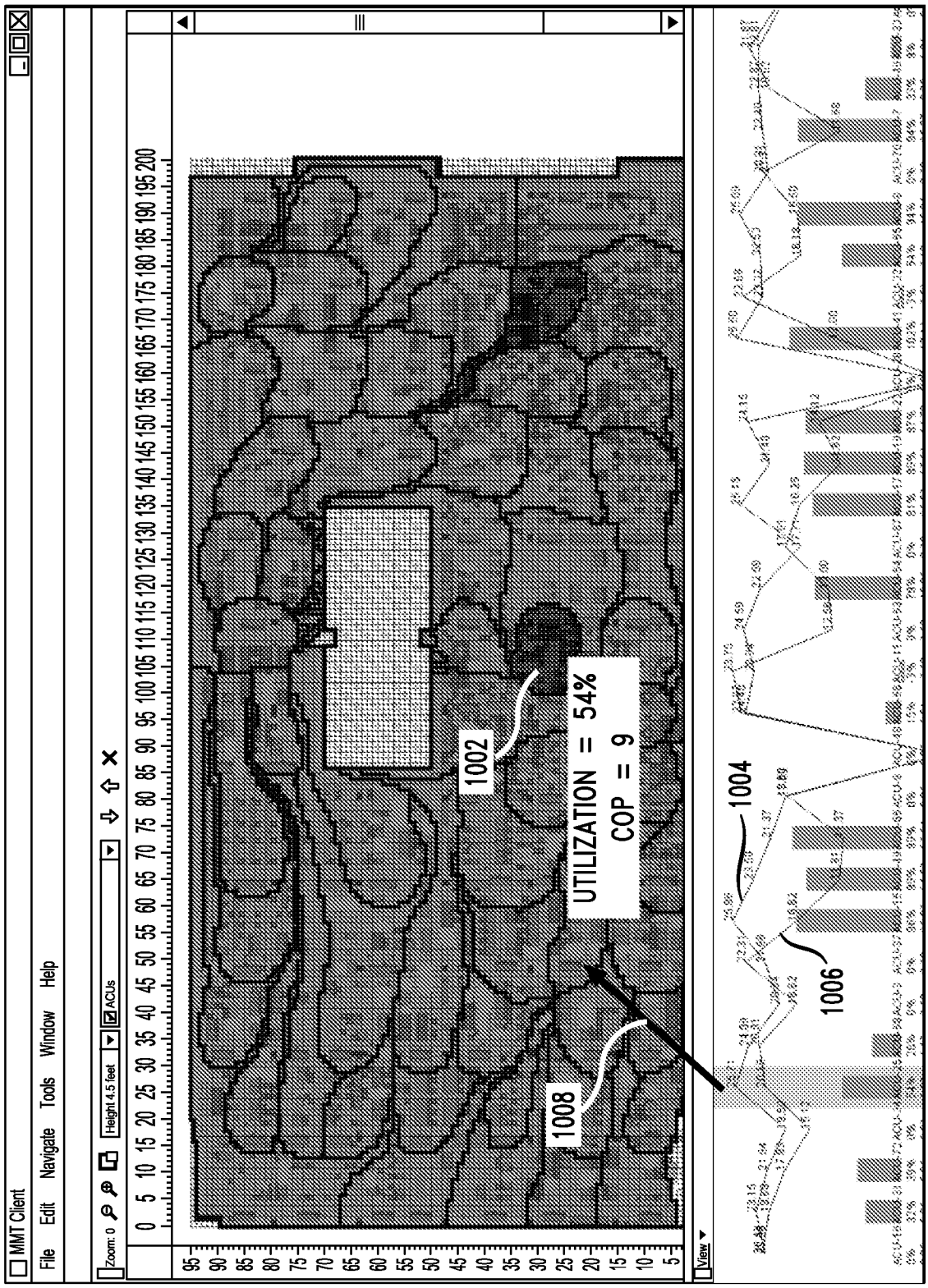

MMT Client
File Edit Navigate Tools Window Help
Zoom: 0
Height 4.5 feet
ACUs
1002
UTILIZATION = 54%
COP = 9
1004
1006
1008
View

FIG. 10

800

FIG. 11

1100

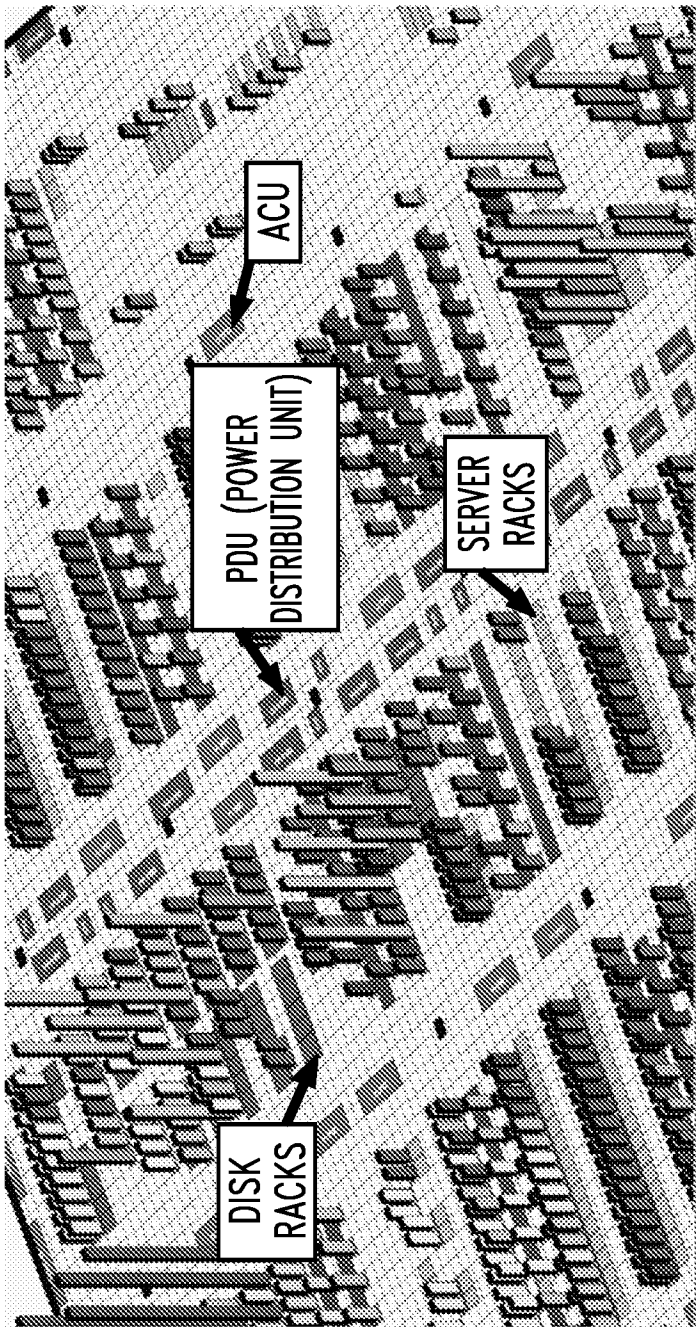


FIG. 12

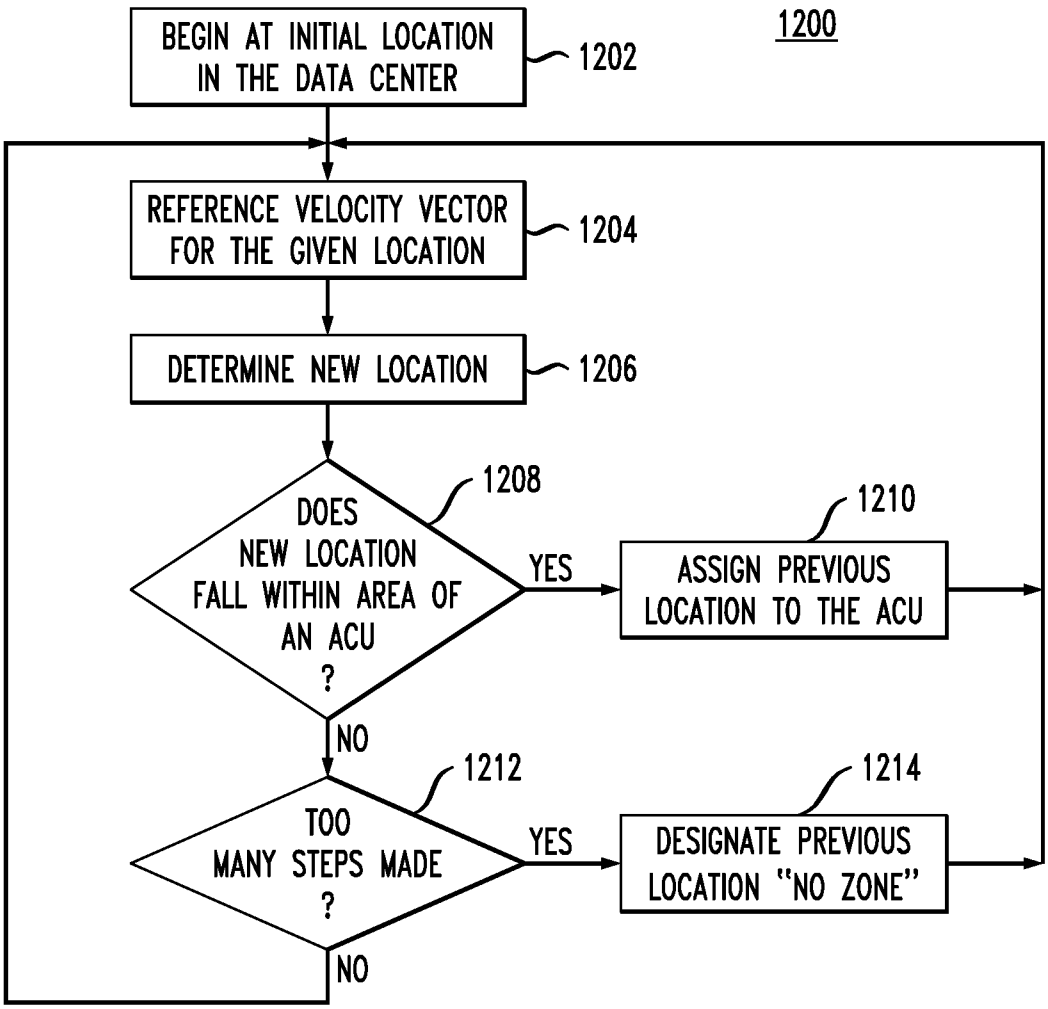
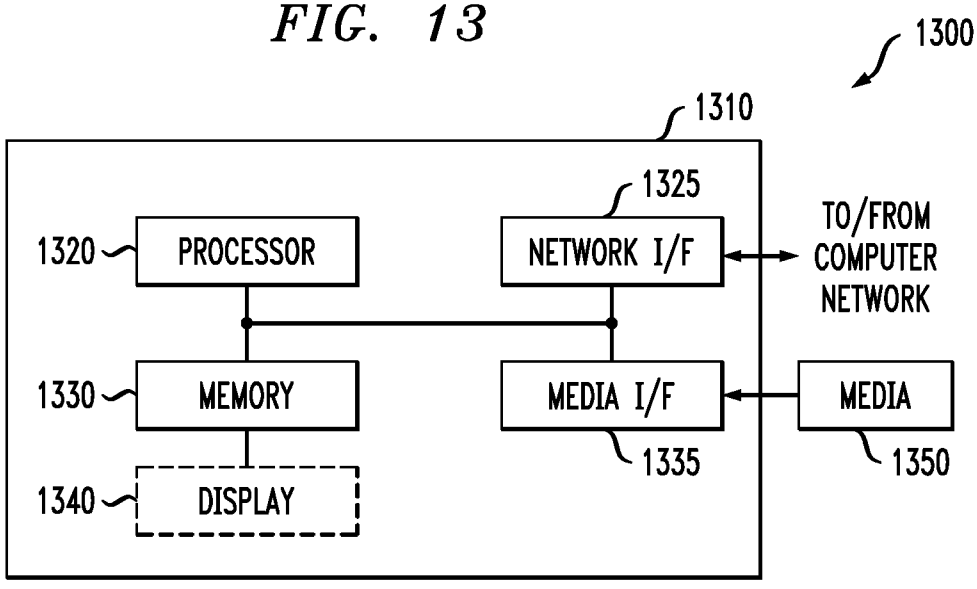

1200
BEGIN AT INITIAL LOCATION IN THE DATA CENTER
1202
REFERENCE VELOCITY VECTOR FOR THE GIVEN LOCATION
1204
DETERMINE NEW LOCATION
1206
DOES NEW LOCATION FALL WITHIN AREA OF AN ACU ?
1208
YES
ASSIGN PREVIOUS LOCATION TO THE ACU
1210
NO
TOO MANY STEPS MADE ?
1212
YES
DESIGNATE PREVIOUS LOCATION "NO ZONE"
1214
NO

FIG. 13


1300
1310
1325
1320
PROCESSOR
NETWORK I/F
TO/FROM COMPUTER NETWORK
1330
MEMORY
MEDIA I/F
MEDIA
1335
1350
1340
DISPLAY

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US2010/044740

A. CLASSIFICATION OF SUBJECT MATTER
IPC(8) - G06F 1/20 (2010.01)
USPC - 703/9
According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
IPC(8) - G06F 1/20 (2010.01)
USPC - 703/9

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)
PatBase

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y	US 2008/0174954 A1 (VANGILDER et al) 24 July 2008 (24.07.2008) entire document	1-18
Y	US 2002/0144473 A1 (SATOMI et al) 10 October 2002 (10.10.2002) entire document	1-18

☐ Further documents are listed in the continuation of Box C. ☐

* Special categories of cited documents:
"A" document defining the general state of the art which is not considered to be of particular relevance
"E" earlier application or patent but published on or after the international filing date
"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)
"O" document referring to an oral disclosure, use, exhibition or other means
"P" document published prior to the international filing date but later than the priority date claimed
"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention
"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone
"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art
"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
06 October 2010	20 OCT 2010
Name and mailing address of the ISA/US Mail Stop PCT, Attn: ISA/US, Commissioner for Patents P.O. Box 1450, Alexandria, Virginia 22313-1450 Facsimile No. 571-273-3201	Authorized officer: Blaine R. Copenheaver PCT Helpdesk: 571-272-4300 PCT OSP: 571-272-7774

Form PCT/ISA/210 (second sheet) (July 2009)